ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 7 N. 4 (2018), 5-16
eISSN: 2255-2863
DOI: http://dx.doi.org/10.14201/ADCAIJ201874516

Improving the adaptability of multi-agent based E-learning systems

Francisco Pinto-Santos, Héctor Sánchez San Blas, Manuel Salgado de la Iglesia, and Xuzeng Mao

franpintosantos@usal.es, hectorsanchezsanblas@usal.es, id00714719@usal.es, xuzengmao@usal.es
University of Salamanca

KEYWORD

multi-agent systems; e-learning; clustering

ABSTRACT

E-Learning is a new learning approach that involves the use of electronic technologies to access education outside of a conventional classroom (Alonso Rincon,). The objective of E-Learning systems is to increase the students' learning skills by providing a customized experience to each system user (Rodrigues, 2013). However, to accomplish this, it is necessary to monitor the continuous changes in the environment, mainly the students' knowledge and skill acquisition. A multi-agent system architecture and a clustering algorithm are proposed for this purpose (as presented in (Rodrigues, 2014) This paper is an extension to the work of (Al-Tarabily, 2018) because it not only monitors changes in the student environment but also in the project environment, increasing the system's adaptability and accuracy.

1. Introduction

E-learning systems provide a dynamic learning environment that allows students to improve their learning skills and to learn faster. The main benefit of this kind of systems is the independence they give to the students, allowing them to flexibly adjust their learning to their routine.

There have been countless proposals in the area of e-learning systems, they offer solutions based on artificial intelligence (AI) ((Vazquez, 2011), (Brusilovsky, 2003)), data-mining techniques ((Chen, 2007), (Romero, 2007)) and fuzzy theory ((Stathacopoulou, 2007), (Huang, 2008)). However, the main drawback of these solutions is their inability to adapt to the dynamic changes that are bound to occur in any learning environment.

Some AI systems such as (Vazquez, 2011) and (Brusilovsky, 2003), and the PSO method have been proposed to solve this issue. PSO is a computational method derived from the cooperative intelligence of insect colonies ((Al-Tarabily, 2018)). By using PSO in dynamic systems, it is possible to track changes and therefore, to respond to them effectively, finding optimal solutions ((Chen, 2016), (GopalaKrishnan, 2016)).

Examples of the use of PSO are: (De-Marcos, 2007), in whose article PSO and a PSO agent were utilized to solve the learning object sequencing problem through automatic sequencing; in (Cheng, 2009) the authors proposed a dynamic question generation system based on the PSO algorithm to solve the problem of selecting questions from a large-scale item bank; (Ullmann, 2015), in which a PSO-based algorithm was developed for Massive Online Open Courses, to form user groups based on the users' level of knowledge and interest. Also, PSO has been used in (Al-Tarabily, 2018), to support educational processes in a dynamic multi-agent system.

This paper presents a multi-agent-based e-learning system, as an extension to (Al-Tarabily, 2018). The architecture designed by (Al-Tarabily, 2018) is composed of five agents and uses the subtractive-DMAPSO algorithm (Kennedy, 1995) to cluster projects and students. Once the clusters are generated, the system executes the matching algorithms to optimally match the student clusters with the project clusters. The first agent used is PCA, it organizes the projects according to the level of skill that a student needs to accomplish them; then the SCA agent clusters the students by their skills; the third one is the SPMA, which maps the student groups formed in the SCA to appropriate projects, according to each group's average level of skill; the fourth is the SSMA which matches students with 'helper' students in order to complement their knowledge; finally, the DSCA, which surveys the environment searching for changes in students' skills and informs the SCA of those changes.

The new proposal enhances the system's adaptability to changes in the environment by adding one more agent, the Dynamic Project Clustering Agent (DPCA), which does the same job as DSCA but instead of keeping track of the students' skills it evaluates the skill level required for the projects. Thanks to this agent it is possible to detect when a project is badly evaluated in terms of the skills required to carry it out, correct it and send these results to the PCA.

By using these agents, it has been possible to map all the students and project correctly. Also, the 'helpers' perfectly complemented the students' skills. Finally, the DSCA and DPCA helped the system to adapt to changes dynamically. This paper is organized as follows: Section 2 reviews the current advances in the field of E-Learning. Section 3 proposes an architecture based on six agents inspired by (Al-Tarabily, 2018). Experimental results are outlined and discussed in 4. Finally, conclusions are presented in Section 5.

2. State-of-the-Art

Nowadays, **E-learning** is one of the most studied areas because it provides efficient learning tools that offer a personalized learning experience to their users (Daradoumis, 2013), (Soller, 2005) and (Kahiigi Kigozi, 2008),. According to (Daradoumis, 2013), one of the most useful E-Learning applications is the Massive Open Online Course (MOOC). Open Educational Resources (OER) are a trend in innovation and offer online distance learning. MOOC facilitates the efficient creation and distribution of knowledge and information. Therefore, it can offer a wide variety of knowledge to a large number of students because it can be accessed remotely via the internet, and is an open and free resource. As stated by the referenced paper , the performance of E-learning systems can be improved with the implementation of a multi-agent system and a similar schema.

(Kamdar, 2018) explain that computer systems have always been centralized, but in recent years there has been a shift from centralized to decentralized systems. The author also sees **Multi-Agent Systems (MAS)** as ideal for constructing decentralized computer systems. As mentioned in (Rivas, 2017) an agent is an autonomous entity capable of collaborating with the nearby environment (Solanki, 2007) defines a MAS as a distributed and coupled network of intelligent hardware and software agents that work together to achieve a global goal. In another work, MAS are described as consisting of several interacting agents that cooperate, coordinate and negotiate, defined in (Wooldridge, 2009). According to (Kamdar, 2018), some of the advantages of this technology are: (i) its distributed architecture based on local information and decision making, (ii) flexibility provided by the "plug and play" capabilities to remove or place agents in the system, (iii) resilience that allows it to quickly respond to failures and fix them.

To build the system proposed by Kamdar, a **data clustering algorithm** was used to improve the efficiency of the systems and to cluster groups of elements with similar features. Most clustering algorithms represent the dataset with feature vectors; these are the features that distinguish one object from another. Only objects with a high level of similarity are grouped in the same cluster; that's why it is essential that the algorithm measure the similarity between two datasets. The Euclidean distance is the most popular method used for this purpose; it obtains the distance between two vectors. In this case, the distance between two data vectors refers to the deviation between students or projects to be clustered. As stated by (Hatamlou, 2013) data clustering is the most important technique in the field of data analysis. For this reason, many researches, like (Hammouda, 2000) have combined it with neural or fuzzy models.

One application of this method is the **Density-Based Spatial Clustering of Applications with Noise (DBSCAN)** algorithm, which is present in several articles, like (Tran, 2013) or (HOU, 2000).

The DBSCAN algorithm has emerged because of the necessity to merge, on the one hand, the students in the SCA agent, and on the other hand, projects in the PCA agent. The DBSCAN is a density-based clustering algorithm proposed by (Ester, 1996). This algorithm has 2 parameters: the first one, epsilon, is a positive number that measures the Euclidean distance between any chosen point and the lowest point of the cluster. From one point, the algorithm expands the cluster to the points which are at epsilon distance. Then, the process is repeated for the new points. Finally, the points that don't belong to a cluster are designated as "noise nodes". The DBSCAN algorithm is used due to its efficiency O(N*log(N)), it's robustness and the need for few specifications. All these advantages fit the clustering needs of this problem. Algorithm 1 and algorithm 2 present a pseudocode of the DBSCAN proposed by (Ester, 1996).

Algorithm 1 DBSCAN Algorithm

```
Input SetOfPoints, Eps, MinPts
// SetOfPoints is UNCLASSIFIED
ClusterId ← nextId(NOISE)
for all Point in SetOf Points do
    if ClusterId = UNCLASSIF IED then
        if ExpandCluster(SetOf Points, Point, ClusterId, Eps, minP ts) then
            ClusterId ← nextId(ClusterId)
```

Algorithm 1 has 3 inputs: the first one, SetOfPoints, is a set of unclassified points; the second one, Eps, is the Euclidean distance; and the last one, MinPts, is the lowest point belonging to a region. Algorithm 1 has a loop that runs through all the points of SetOfPoints. If the point is not in any cluster (UNCLASSIFIED), algorithm 1 expands the cluster from that point. If the expansion is satisfactory, a cluster is generated. Then, the search for the next cluster beings with the remaining points that have not been classified.

Algorithm 2 ExpandCluster

```
    Input SetOfPoints, Point, ClId, Eps, MinPts
    Output Boolean
seeds ← SetOfPoints.regionQuery(Point,Eps)
// returns the Eps-Neighborhood of Point in SetPoints as a list of points.
if seeds.size < MinPts then
    SetOf Point.changeClId(Point, NOISE)
    ReturnFalse
else
    SetOf Points.changeClId(seeds, ClId)
    seeds.delete(Point)
    while seeds! = Empty do
        currentP ← seeds.first()
        result ← SetOfPoints.regionQuery(currentP, Eps)
        if result.size >= MinPts then
            for all resultP in result do
                if resultP .ClId IN (UNCLUSSIFIED, NOISE)
                    then if resultP .ClId = UNCLASSIF IED then
                        seeds.append(resultP )
                    SetOf Points.changeClId(resultP,
        ClId) seeds.delete(currentP )
    returnTrue
```

Algorithm 2 has the following inputs: SetOfPoints, the Point that is going to be expanded; the identification of the new cluster (ClId); Eps; and MinPts. In the first place, algorithm 2 gets the points (seeds) that are at a distance Eps of the Point. These points are obtained with the method RegionQuery of SetOfPoint. If the number of points is less than MinPts, algorithm 2 becomes false and the cluster is not created. Otherwise, all points of seeds are assigned to the cluster by changing UNCLASSIFIED to ClID. Then, while the seed is not empty, algorithm 2 takes a point of seeds and gets the points that are at a distance Eps of this point. With these new points, those that are UNCLASSIFIED or have NOISE are added to the cluster. If they are UNCLASSIFIED they will also be added to the seeds.

Another issue that must be solved are the mapping algorithms used by the Student-Project Matching Agent (SPMA) and the Student-Student Matching Agent (SSMA):

1. **Mapping algorithm for SPMA**: This algorithm is used to match each student cluster with a suitable project. For this purpose, a list of student clusters and a list of project clusters is necessary.

   For each student cluster, the students' average skill level is calculated. After that, the algorithm searches for a project cluster whose Euclidean distance is shorter than epsilon (which is a factor based on the experiments that have been performed to obtain the optimal results). This means that the skills in the project cluster are similar to the skills in the student cluster. The algorithm then finds in the project cluster a suitable project for the student cluster.

   Algorithm 3 contains a pseudocode for the mapping.

Algorithm 3 Student-Project Mapping Algorithm

```
Input studentClusters:StudentCluster[], projectClusters:ProjectCluster[], EPSILON:float,
DELTA:float
Output dictionaryResult:Dictionary<StudentCluster,Project>
for all studentCluster in studentClusters do
    Do average of each skill of all the student of the studentCluster and save it in averageSkills
    projectFound ← false
    currentIteration ← 0
    while !projectFound
    do
        for all projectCluster in projectClusters do
            currentIteration ← currentIteration + 1
            euclideandistance ← euclideanDistance(projectCluster[0].skills, averageskills)
            if EPSILON * currentIteration >= eclideanDistance
                then for all project in projectCluster do
                    projectFound ← true
                    projectSkills ← project.skills
                    for i ← 0 ; i < projectSkills.size; i ← i + 1 do
                        difference ← projectSkills[i] − averageSkill[i]
                        if difference < 0 or difference > DELTA * currentIteration then
                            projectFound ← false
                            break
                    if projectFound = true then
                        dictionaryResult.add(studentCluster, project)
                if projectFound = false then
                        projectFound ← true
                        dictionaryResult.add(studentCluster, projectCluster[0])
                    break
```

2. **Mapping algorithm for SSMA**: The mapping algorithm for SSMA is used to match each student cluster with a helper. To this end, a list of student clusters and a list of helpers is needed.

For each student cluster, the average skill level is calculated. After that, a suitable helper who has slightly better skills than the average skills of the student cluster is chosen. Algorithm 4 presents a pseudocode for the mapping.

Algorithm 4 Student-Student Mapping Algorithm

```
    Input studentClusters:StudentCluster[], helpers:Student[], DELTA:float
   Output dictionaryResult:Dictionary<StudentCluster,Student>
for all studentCluster in studentClusters do
    Do average of each skill of all the student of the studentCluster and save it in averageSkills
    helperFound ← false
    currentIteration ← 0
    while !helperFound
    do
        for all helper in helpers do
            helperFound ← true helper-
            Skills ← helper.skills
            for i ← 0 ; i < helperSkills.size; i ← i + 1
                do if averageSkills[i] < 100 − DELTA
                then
                    difference ← helperSkills[i] − averageSkill[i]
                    if difference   <   DELTA/currentIteration or difference   >
                                        DELTA *
(currentIteration + 1) then
                        helperFound ← false
                        break
                    else if helperSkills < DELTA * currentIteration then
                        helperFound ← false
                        break
            if helperFound = true then
                dictionaryResult.add(studentCluster,
                helper) break
```

The last task of the proposed system is to **capture the environmental changes**, in other words, to keep track of the students' learning curve and to evolve the characteristics of the projects. This enhances the system with greater adaptability, which translates into faster knowledge and skills acquisition among students. As proposed in (Al-Tarabily, 2018), two agents are used to capture the environmental changes: the PCA and the DSCA, which capture changes in the students' knowledge and skills but do not keep track of the changes in the data of the projects. However, the system proposed in this paper takes this aspect into account because not all students approach the projects in the same way. The main contribution of this paper is the addition of one more agent, the DPCA. Moreover, the behaviour of the DSCA and PCA agents is changed. The PCA is the only clustering agent, and the DSCA and DPCA capture changes in the projects and students respectively.

3. Proposed architecture

To improve the performance in the learning process, the presented architecture, is an alternative to the multi-agent system proposed in (Al-Tarabily, 2018), adding an agent (the DPCA), to capture the data changes in the projects. This architecture consists of six agents, of whom five come from (Al-Tarabily, 2018) (Project Clustering Agent (PCA), Student Clustering Agent (SCA), Student-Project Matching Agent (SPMA), Student-Student Matching Agent (SSMA) and the Dynamic Student Clustering Agent (DSCA)). Also, a new agent has been added to this architecture, called the Dynamic Project Clustering Agent (DPCA) which improves the system, allowing it to perceive changes in projects. Due to the introduction of this agent, the communication behavior of the rest of the agents is modified, as illustrated in the AUML diagram in Figure 2.. According to (Al-Tarabily, 2018), to manage the data, the system divides the information into two types of profiles: student and project. The profiles information is obtained with the data introduced by the teachers. However, they should be dynamic, in other words, the profiles' information should be updated when the environment changes. This is because the profiles have two types of data: static and dynamic. The former are collected from the users through a questionnaires and from the information introduced by the teachers. The latter are collected from the users' interaction within the system, and they change continuously the students obtain knowledge and improve their skills, for the most part by completing projects.

The system has been implemented in JAVA, specifically in the JADE environment for two reasons. Firstly, it provides many functionalities and facilities for the implementation of a multi-agent system, including scalability. Secondly, this research has been developed as part of the advanced programming course of the University of Salamanca, in which we used these technologies. Below we describe the behavior of each agent:

1. **PCA**: is a reactive agent, which receives messages from the DPCA when some project profile changes, and then re-clusters the project profiles' into homogeneous groups using their attributes. PCA uses the DBSCAN algorithm to efficiently cluster the projects on the basis of their level of difficulty. The clustering algorithm is described in more detail in (Ester, 1996).
2. **SCA**: The accurate clustering of students according to their abilities is essential and has a significant impact on improving the performance of the e-learning system. This is because, if the student clusters are not homogeneous, the good students' learning process can be slowed down by poorly performing students; and the teacher's workload can increase. Like the PCA, the SCA agent uses the DBSCAN algorithm presented in (Ester, 1996) to cluster the students.
3. **SPMA**: The different student and project groups produced from the SCA and PCA are utilized by SPMA. This agent maps student groups to projects according to the level of skill of the student group and the project requirements. This average skill level is calculated on the basis of the static data and the students' previous scores within the system, so when astudent cluster finishes a project the information is updated. The algorithm executed by the SPMA agent is described in Subsection 2.
4. **SSMA**: This agent records the student's knowledge, learning style, time availability and preferences, and sustains a dynamic student profile. Thanks to this profile the SSMA recommends the best student to help in the project of another student. For example, if a student has a doubt regarding their project, the agent suggests a helper that has a considerable level of knowledge about the project. This function of the SSMA is based on DBSCAN (Ester, 1996) . The algorithm executed by this agent is described in Subsection 2.
5. **DSCA and DPCA**: These two agents have two tasks:
   - The first task is to record any changes in the students' and projects' skills.
   - The second task is to request the corresponding clustering agent (The DPCA communicates with the PCA and the DSCA communicates with the SCA).

The second task is performed once a project has completed, because the skills in the student group are improved and the required skill level of the project must be readjusted.

Communication between agents is shown in Figure 1. All the agents in the system are reactive, that is to say, they proceed to carrying out a task when they receive an external stimulus and send a confirmation to the next agent once they have completed it. The DPCA and the DSCA agents react to information changes in the students' and projects' profiles respectively, they then send a message, with a request to PCA or SCA respectively, and wait for a message with a confirmation of task completion. When PCA, SCA, SSMA and SPMA receive a request, they do their clustering tasks (in the case of the two first) or their mapping tasks (in the case of the last two) and then send a message with a confirmation to the sender that they have completed the task. After that, in the case of PCA and SCA, they warn SPMA, or to SPMA and SSMA respectively.

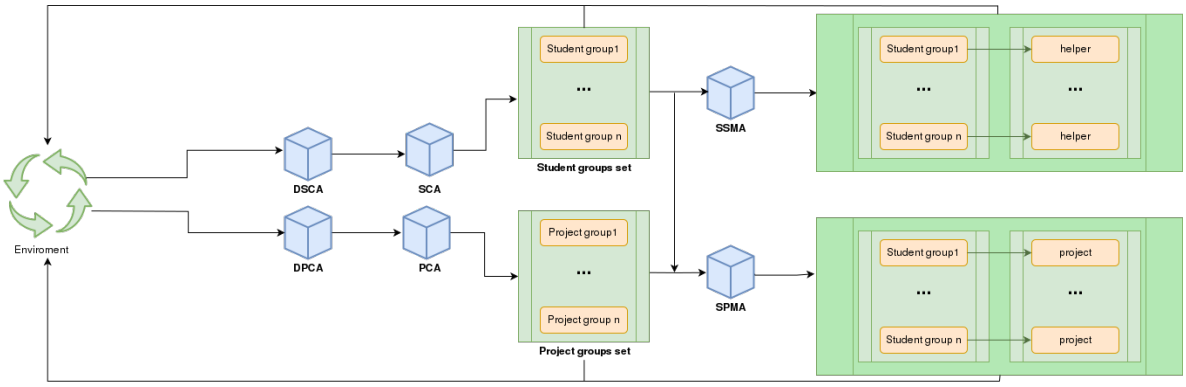

Figure 1: System operation

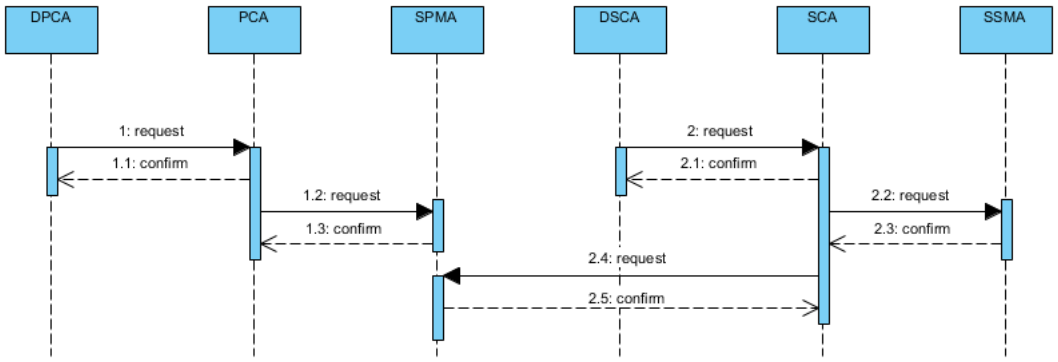

Figure 2: AUML system operation

4. Experimental results

An experiment has been performed to verify the efficiency of the developed system. The test was an extension to the one described in (Al-Tarabily, 2018) as the system was provided with an additional agent for capturing changes in the project data. The experiment began by doing two surveys; Google was used to collect, on the one hand, information about the skills and abilities of a hundred students (the names of the students who answered the survey are not listed for privacy reasons), and on the other hand, information about a hundred projects and their skill level (these projects were proposed by different teachers, whose names are also not listed for privacy reasons).With this information, one project bank is constructed with a hundred projects and one student bank with a hundred students. In this scenario (both students and projects), three abilities are analyzed. In particular, the skills that have been chosen are those necessary for the development of graphical user interfaces, a subject studied in the University of Salamanca (knowledge about C, Java and Graphic Design).

The students' skills and the projects' skill level, were clustered by SCA and PCA respectively. The clustering process depends on the Euclidean distance. The procedure began choosing a student and adding to the cluster all the students who are at a distance equal or lower to the Euclidean distance. After this, the process is repeated by each of the scholars included until there are no students in a distance equal to the Euclidean distance. Table 1 shows the characteristics of both the student (S1) and project (P1) banks. For both of them, it's showed the number of instances in each of them as the number of skills they have and the number of clusters we got from the SCA and PCA.

Table 1: Characteristics of the projects and student's banks, and the number of clusters obtained

Item bank	Number of instances	Number of attributes	Number of clusters
P1	100	3	9
S1	100	3	13

After this, the SPMA assigns the appropriate projects to the students according to their characteristics and to the skills required by the projects. In this way, the SPMA uses the Student-Project Mapping Algorithm. It crosses all the clusters of students, taking the average of their skills, to be able to find a project with a greater difficulty to their skills. In this way, by carrying out some of the projects in the cluster, the students will be able to improve their knowledge. An example of the result of this process can be found in Table 2 where it's showed the values of a student (Student's Skills) and a project (Needed Skills) for each of it's skills. As can be seen, the values in both columns are similar.

Finally, the groups of students are matched with 'helpers students', who compensate their skills. In this way, the SSMA uses the Student-Student Mapping Algorithm. It crosses all the clusters of students, taking the average of their skills, to be able to find a helper with a greater knowledge. Through the appropriate assignment of the helper, the students of the cluster will be able to develop the projects assigned to them with a help that allows them to learn and advance with the work. In this way we get Table 3, which shows two students (Student 1 and 2) from a group with their correspondent assistant (Helper) and the skills of both of them. As can be observed, the helper assigned to the group has slightly higher skills than the students of the group.

Table 2: Example of the characteristics of a student with the necessary characteristics to carry out the assigned project

	Student's skills	Needed skills
Skill 1	91.0	93.0
Skill 2	12.0	18.0
Skill 3	88.0	93.0

Table 3: Example of the characteristics of two students with the characteristics of the assigned helper

	Student 1	Student 2	Helper
Skill 1	50.0	49.0	60.0
Skill 2	54.0	45.0	58.0
Skill 3	91.0	82.0	97.0

After finishing this first iteration of the experiment, the DSCA and the DPCA come into action; they detect changes in students and projects respectively and send the results to the PCA and SCA agents to repeat the process. A diagram illustrating the communication between the agents is presented in Figures 3, 4 and 5.

After the above-mentioned agent activities, the students who took the survey were clustered by the SCA in different groups according to their abilities, and the projects were organized in the same way by the PCA. After that, the projects were assigned to the groups of students by the SPMA, which mapped the different groups to the projects based on the projects' average level of skill and the groups' average level of skills. Finally, this data is sent to SSMA, which recommends the best student to help (called 'helpers' in the diagrams) in the project of another student in order to compensate for the student's lack of a certain skill. All this information is provided dynamically by two agents: DSCA, and the one that we add to the original work, the DPCA. The former agent searches for changes in the students skills and then informs the SCA about those changes, while the second one does the same task but with the changes in projects and informs the PCA. This experiment, mapped all the students and projects correctly, the level of the students was well assessed and all the students were placed in groups that corresponded with their abilities; the projects were equally well organized by the level of difficulty. Also, the 'helpers' perfectly complemented the students' skills so the result was that all the students ended up satisfied with the groups they were in and the project they were assigned to do and with the 'helpers' they worked with. In other words, we achieved the desired results.

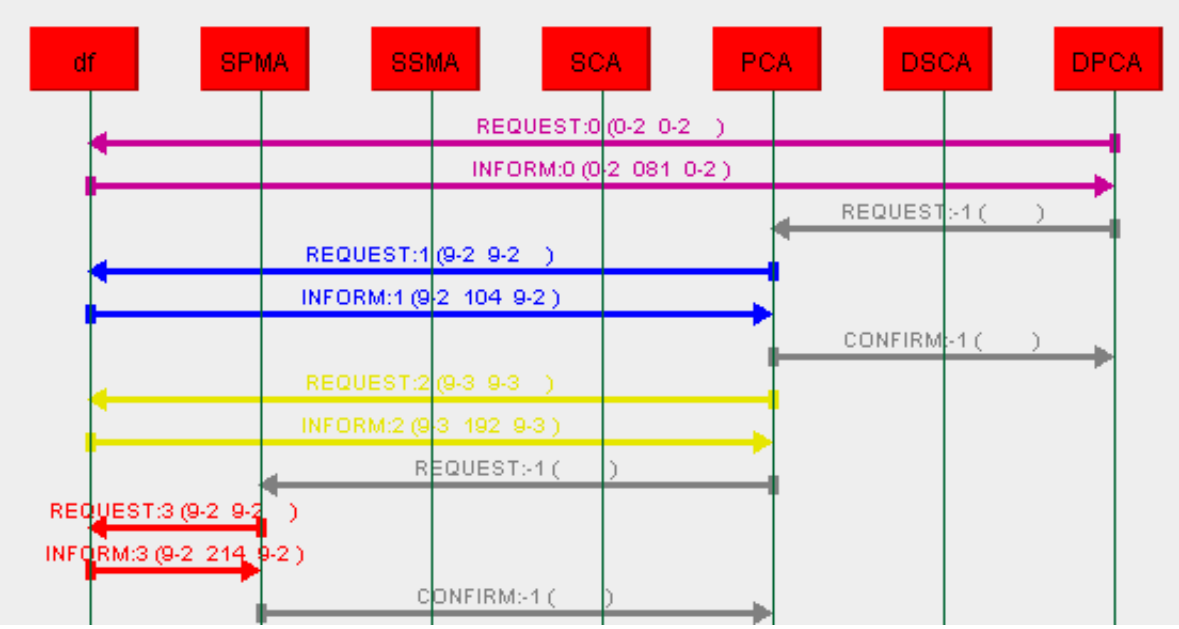

Figure 3: Experimental scenario in JADE, the first one showing changes in students' skills, the next one in projects and the last one in students again. Part 1.

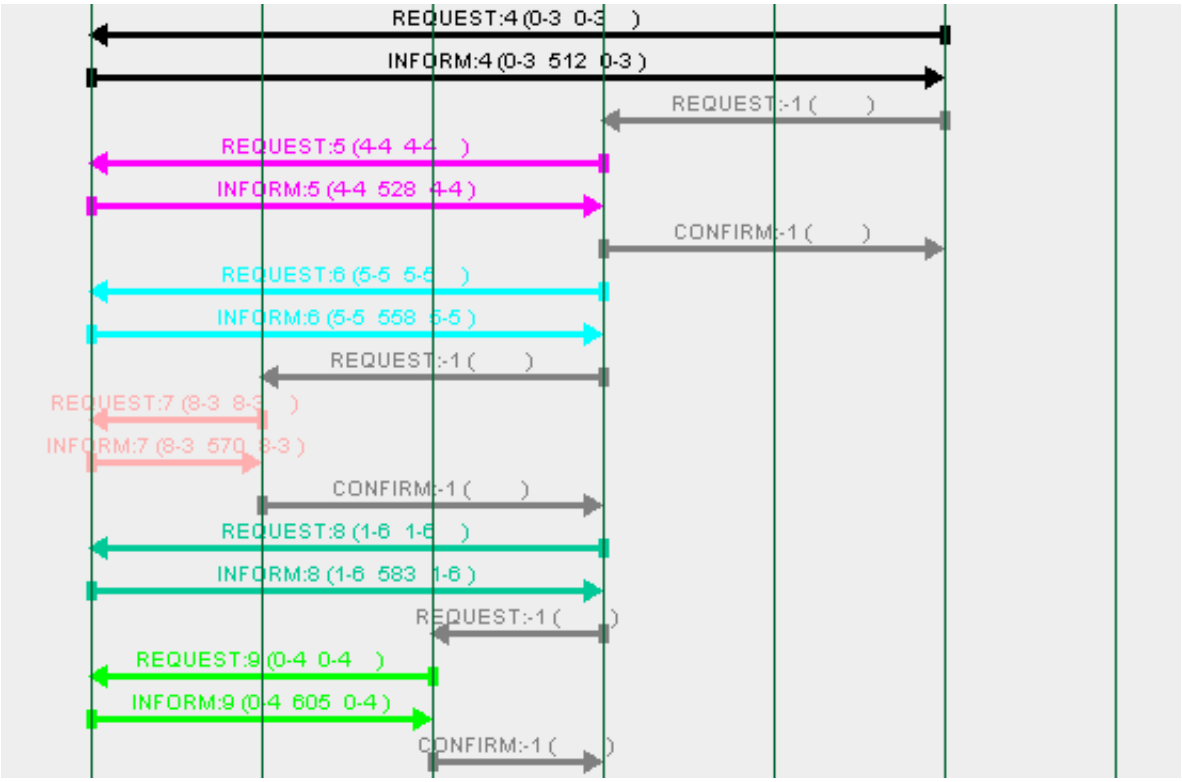

Figure 4: Experimental scenario in JADE showing changes in students' skills, the next one in projects and the last one in students again. Part 2.

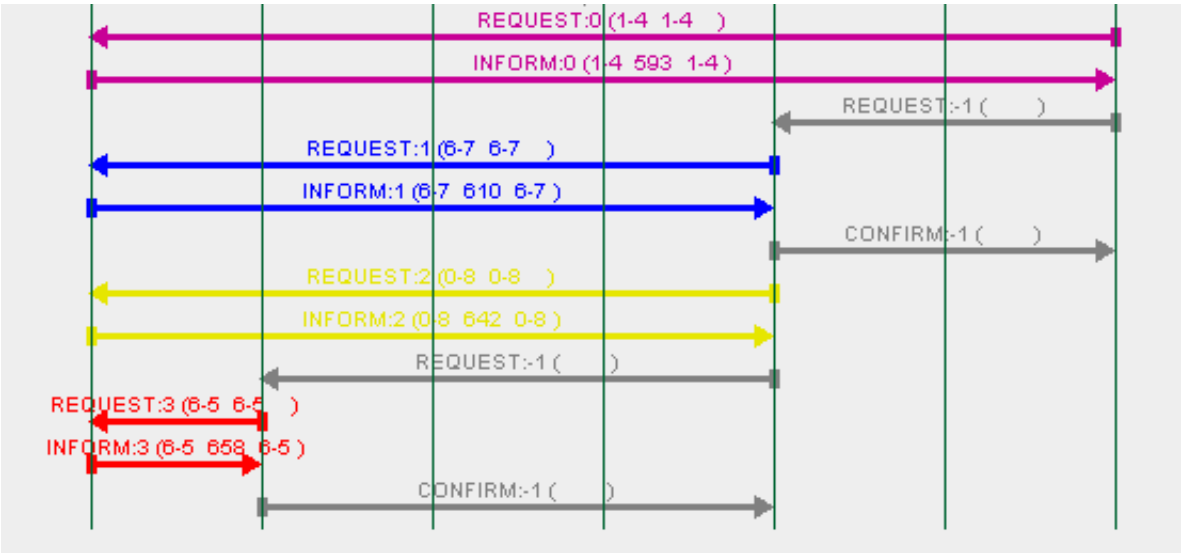

Figure 5: Experimental scenario in JADE showing changes in students' skills, the next one in projects and the last one in students again. Part 3.

5. Conclusions

In this paper, an extension to the work of (Al-Tarabily, 2018) has been proposed by adding a new agent: the DPCA, which searches for changes in the projects' skills and abilities and informs the PCA of those changes. The architecture proposed for this alternative, accomplished in the JADE environment and implemented in JAVA, consists in five agents through which it is possible to carry out the desired process. The PCA and SCA cluster projects and students according to their skills, using the DBSCAN algorithm. These clusters are linked to the SPMA, which matches a group of students with a group of projects whose difficulty level is corresponds to the abilities of the students. After that, the SSMA assigns a 'helper' student to each the cluster of students in order to help them. Finally, the DSCA and DPCA, the agent proposed as an addition to (Al-Tarabily, 2018), which detect the changes in students's abilities and in the projects, respectively. These agents inform the clustering agents that a new iteration must take place. The results of the test demonstrate that the proposed extension to the system developed by (Al-Tarabily, 2018) is valid. As evidenced by the test, all the students' were correctly clustered by their level and so were the projects. Moreover, the student groups and the test groups were well matched, corresponding in the ability level. Finally, the assignment of helpers was a success because the person assigned as a 'helper' had better skills than the group of students to whom he was assigned and could help them to improve. Finally, in a future research we plan to perform the test with a larger sample size and considering more factors, such as students' personal preference to belong to a certain group. The mapping and clustering algorithms could also be optimized and other alternatives could even be explored.

6. References

Al-Tarabily, A. A. M. I., F. Abdel-Kader, 2018. Optimizing Dynamic Multi-Agent Performance in E-Learning Environment.

Alonso Rincon, G. P. C. R., Prieto Tejedor. Collaborative learning via social computing. Frontiers of Information Technology Electronic Engineering.

Brusilovsky, P., 2003. Adaptive and intelligent technologies for Web-based education. Volume 13, pages 159–172.

Chen, C. S., Chen, 2016. Ontology-based adaptive dynamic e-learning map planning method for conceptual knowledge learning. Volume 11, pages 1–20.

Chen, H., Hsieh, 2007. Mining learner profile utilizing association rule for Web-based learning diagnosis. Volume 33, pages 6–22.

Cheng, H., Lin, 2009. Dynamic question generation system for Web-based testing using particle swarm optimization. Volume 36, pages 616–624.

Daradoumis, X. C., Bassi, 2013. A review on massive e-learning (MOOC) design, delivery and assessment. De-Marcos, M. G., Pages, 2007. Competencybased learning object sequencing using particle swarms. Volume 2, pages 111–116.

Ester, S. X., Kriegel, 1996. A density-based algorithm for discovering clusters in large spatial databases with noise.

GopalaKrishnan, S., 2016. A hybrid PSO with Naïve Bayes classifier for disengagement detection in online learning. Volume 50, pages 215–224.

Hammouda, K., 2000. A Comparative Study of Data Clustering Techniques. page 1.

Hatamlou, 2013. Black hole: A new heuristic optimization approach for data clustering. Volume 222, pages 175–184.

HOU, C. F. H., ZHOU, 2000. Approaches for Scaling DBSCAN Algorithm to Large Spatial Databases. Volume 15.

Huang, H. J. K., Chen, 2008. Standardized course generation process using dynamic fuzzy Petri nets. Volume 34, pages 72–86.

Kahiigi Kigozi, H. T., Ekenberg, 2008. Exploring the e-Learning State of Art. Volume 6, pages 77–88. Kamdar, K., Paliwal, 2018. A State of Art Review on Various Aspects of Multi-Agent System. Volume 27, page 15.
Kennedy, E., 1995. Particle Swarm Optimization.
Rivas, R., Chamoso, 2017. An Agent-Based Internet of Things Platform for Distributed Real Time Machine Control. Pages 1–5.
Rodrigues, F. R., Gonçalves, 2013. E-Learning platforms and E-learning students: Building the bridge to success. Volume 1, pages 21–34.
Rodrigues, F. R., Gonçalves, 2014. Developing multimodal conversational agents for an enhanced e-learning experience. Volume 3, pages 13–26.
Romero, V., 2007. Educational data mining: A survey from 1995 to 2005. Volume 33, pages 135–146. Solanki, S., Khushalani, 2007. A multi-agent solution to distribution systems restoration. Volume 22, pages 1026–1034.
Soller, J. M., Martinez, 2005. From Mirroring to Guiding: Review of State of the Art Technology for Supporting Collaborative Learning. Volume IJAIED, pages 261–290.
Stathacopoulou, S. M., Grigoriadou, 2007. Monitoring students' actions and using teachers' expertise in implementing and evaluating the neural network-based fuzzy diagnostic model. Volume 32, pages 955–975.
Tran, D., Drab, 2013. Revised DBSCAN algorithm to cluster data with dense adjacent clusters. Volume 120, pages 92–96.
Ullmann, C. C. d. A., Ferreira, 2015. Formation of learning groups in cMoocs using particle swarm optimization. pages 3296–3304.
Vazquez, G.-A. M., Ramirez, 2011. Designing adaptive learning itineraries using features modelling and swarm intelligence. Volume 20, pages 623–639.
Wooldridge, 2009. Introduction to Multi-Agent Systems.