International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V72I9P105

Volume 72 Issue 9, 58-66, September 2024

Original Article

Development of a Multiagent Model based on the Development of Distributed Applications

Abdellah Amine[1], Rachid Ait Daoud[2], Fatiha Ait Yacine[3]

[1,2]Systems Engineering Laboratory (LGS), Higher School of Technology(EST_FBS) Sultan Moulay Slimane University, Fkih Ben Salah, Morocco.
[3]Innovation Laboratory for Mathematics, Applications and Information Technology, Sultan Moulay Slimane University Beni Mellal, Morocco.

[1]Corresponding Author : a.amine@usms.ma

Received: 10 April 2024 Revised: 13 July 2024 Accepted: 02 August 2024 Published: 28 September 2024

Abstract *- This strategy aims to automate particular education-related administrative functions. The goal of this automation is to optimize the work carried out in these services by drawing on the theories of Distributed Artificial Intelligence (DAI) and Multiagent Systems (MAS). In order to do particular tasks, this multiagent application integrates entities known as agents that collaborate and communicate with one another. The Java Agent Development Framework, or JADE middleware, serves as the foundation for the system and is used to develop and manage agents. We have evaluated the multiagent system concept using private data from an experiment we ran with university students.*

Keywords *- Artificial Intelligence (AI), Distributed Artificial Intelligence (DAI), Multiagent System (MAS), Agent, UML, JADE.*

1. Introduction

The implementation of computerized management is becoming increasingly prevalent in academic institutions, as it facilitates the streamlining and optimization of various tasks and procedures. Indeed, numerous universities are adopting computerized management systems to ensure the accurate and efficient fulfillment of student needs. These traditional procedures are lengthy, costly, and less secure than the time and effort expended by students and administrators [1, 2]. The following section will address the procedures and problems associated with the provision of educational services.

- Manual filing of documents
- Limitation of staff availability
- Difficulty in processing requests.
- Less security.
- There is a lot of work to do to control the number of documents.

In light of the aforementioned considerations pertaining to the enhancement of informational resources within our academic institution, we have contemplated the automation of select academic services. The objective is to create a distributed application based on intelligent agents [3] that can meet these requirements. Our Multiagent System (MAS) [4] is an organized set of agents. It consists of one or more organizations that structure the rules for cohabitation and collective work among agents. An agent must be able to communicate, cooperate, and coordinate with other agents [5].

At present, numerous agent platforms facilitate the development and management of multiagent systems. The selection of an appropriate agent platform is contingent upon the specific objective of the intended application. The work already carried out is a distributed application based on the object-oriented paradigm, in addition to a design of a primary SMA Model. The model developed is a more dynamic, extensible, and adaptable SMA Model that fits the architecture of our university information system.

2. Related Work

Since the 1950s, AI has been structured around two main currents:
1. Symbolic AI: which is based on a "symbol" operating system, inspiring technologies like the "expert system" based on a set of rules, for example.
2. Machine Learning: This is an automatic learning system based on data exploitation, imitating a neural network.

Artificial Distributed Intelligence (DAI) [6] was born in the early 80s with the goal of addressing the shortcomings and enriching the classical approach of AI by proposing the distribution of expertise among a group of agents, not subject to centralized control, capable of working and acting in a common environment and resolving any conflicts. In short, DAI is interested, among other things, in modeling intelligent behaviors that result from cooperative activity among multiple agents.

The concept of intelligence here symbolizes the ability of an agent to carry out cooperative, rational, dynamic, and flexible behavior in a moving and often non-deterministic environment. Autonomous agents and Multiagent Systems (MAS) offer a fresh approach to analysis., designing and implementing sophisticated applications because they are part of the field of DAI (Distributed Artificial Intelligence) and also benefit from other disciplines such as cognitive science, sociology and social psychology. Early on in the field of Distributed Artificial Intelligence (DAI), research was primarily concerned with building and developing systems that would demonstrate its originality. From a philosophical perspective, the individual architectures of agents were the focus of attention [7]. In recent years, attention has slowly shifted to the social aspects of knowledge and action [8] with an emphasis on the multiagent systems (MAS) approach. These offer two points of view for modelling a complex system:

- Agent: autonomous software entity capable of reasoning and interacting through exchanges of knowledge [9, 10] with the other agents in the system in order to satisfy its own goals or the overall goals of the system;
- Society: the set of representations and mechanisms required to manage the management of interactions and the organisation or structuring of agents in systems [11].

The purpose of this section is to take stock of software agents, multiagent systems, and their architectures [12], as well as to understand how they manage to cooperate and coordinate their actions to achieve the objectives assigned to their organization or society while remaining autonomous. The different modes of communication between agents[13, 14] and the most well-known communication languages in this field were also discussed. This will allow us to propose multiagent model architectures to integrate them into our university information system.

3. Material, Software and Method

In this work, we used an HP laptop with an Intel core i5@2.6 GHz processor with 16 GB of DDR4-3200 MHz RAM.

In the choice of architecture and tools, the following criteria are taken into account:

- The application must be based on a powerful distributed architecture;
- Development tools should be open source;
- The application must be consulted with a web browser;

We therefore adopted in our project the 3-third distributed architecture based on the J2EE (Java Enterprise Edition) platform and the MVC pattern, which allows break down of the application into three layers (levels), comprising a J2EE web server, a J2EE business server and a database server to have a division between the different parts of the application.

JEE (Java Enterprise Edition) facilitates enterprise application development by providing a runtime environment and API components. The MVC model (Model, View, Controller) is a design pattern used to cut literally the application into separate layers for the organization of the code:

A controller is implemented as a Java Servlet, The model consists of the implementation of the business logic of the application;

The view is implemented via a JSP (Java Server Pages). Servlets implement the controller layer of the MVC model to process requests from a client web browser and generate a dynamic output that is HTML for the client. JSP, implement the View layer of the MVC and present web pages.

Apache Tomcat: the application server that manages servlets NetBeans is the Integrated Development Environment (EDI), which we have adopted to make our JEE-based application. Oracle SQL Developer represents the environment we used to connect to DB Power AMC: we used this software in UML modeling and others.

4. Proposed Architecture

A distributed model based on MAS [15] clearly shows its dynamism and flexibility in adapting to its integration environment. It also presents the agents used in this model, the communications and interactions between these agents, and their these agents, as well as their deployment environments.

4.1. The Multiagent Platform JADE

The JADE platform (Java Agent Development Framework) [16] is an agent development environment that has been entirely implemented in Java. The objective of the JADE platform is to facilitate the development of Multiagent Systems (MAS) by providing a comprehensive set of services and agents, including a naming service, a Yellow Pages service, a parsing service, and a protocol interaction library, all of which are readily available for use. JADE is a platform that is based on the specifications set forth by the Foundation for Intelligent Physical Agents (FIPA) [17]. It comprises three primary modules, all of which are required standards set forth by FIPA. The aforementioned modules are initiated upon each platform startup.

- The Agent Management System (AMS):
- The intelligence agentDirector Facilitator (DF)
- The Agent Communication Channel (ACC)

4.2. Interaction between Agents

One of the most important properties of an agent in an SMA is its ability to interact with other agents. These interactions are generally defined as any form of action performed within the system that has the effect of modifying the behavior of another agent. They allow agents to participate in satisfying an overall goal [18]. These interactions take two forms (direct and indirect).

4.3. Application Types for Multiagent Systems

The agent-oriented paradigm comes into its own when it comes to modeling systems that are open to extensibility, flexibility, and self-adaptive. Indeed, the agent-oriented paradigm seeks to connect software resources to create an integrated system capable of solving complex problems and self-adapting to different situations.

Generally speaking, SMA application types are broken down into three main classes [19]:

- Modelling of real-world phenomena
- Program design
- Distributed problem-solving

The main types of agent are [20]:

- The Dummy agent:
- The Sniffer Agent
- The Introspector agent:

The Figure 2 shows the architecture of this "Intelligent Schooling service" MAS model. Our MAS "Intelligent Schooling" model is a collection of persistent, autonomous, cooperative and mobile agents [21] operating in a distributed environment.

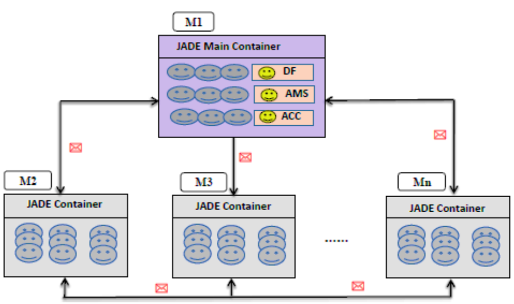

Fig. 1 Architecture of the JADE Platform

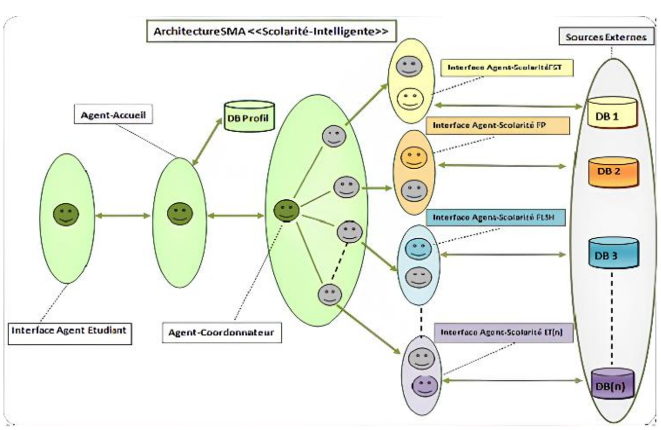

Fig. 2 MAS model architecture

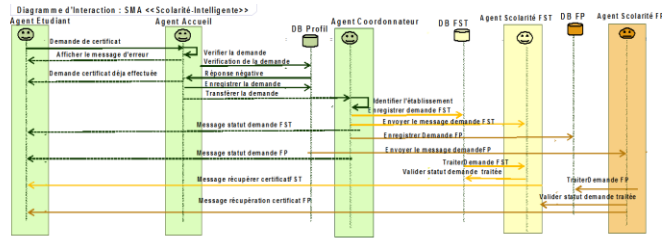

Fig. 3 MAS "Intelligent Schooling" interaction diagram

The automated enrollment processes start with student requests and continue through the generation of documents by the enrollment agents for all the university establishments. This model is made up of the following agents.

Three stationary agents: "Student Agent", "Reception Agent", and "Coordinator Agent". "Agent-Coordinator"; "Schooling Agents", whose number depends on the number of establishments in the university Mobile agents, one per institution;

Each agent is represented in his or her "Container" environment. A DB profile database containing the student's profile; A database for each institution to record applications. The interaction diagram in Figure 3 shows the different agents of the MAS "Intelligent Schooling" model, as well as their interactions and communications. It also describes all the stages in a request, from the beginning to the retrieval of the certificate. This model is adaptable to any number of institutions.

Every request made by a student via the "Student Agent" interface passes through the Agent" interface and goes through all the stages of the "Reception Agent" and "Coordinator Agent». The latter identifies the name of the school and deploys a mobile agent to the school, which moves to the 'container' environment of the corresponding agent's 'container' environment to communicate the request and record it in its database.

So from the "Coordinating Agent", mobile agents are deployed who move to the school environments, and the number of schools is variable. In this diagram, we have only shown two establishments for clarity. This diagram provides a concrete illustration of the advantages and new modelling solutions offered by multiagent systems. In fact, they offer the possibility of directly representing individuals, their behaviour and their interactions [22], thanks to the agents which interact between them in an autonomous way and modular way and their interactions, thanks to agents that interact with each other in a modular way

5. Results and Discussion

5.1. SMA Model Agents

The information system of our university is distributed, and the development of distributed, open, and adaptable applications to its integration environment requires means that can provide solid solutions and that are able to meet the requirements.

The distribution property requires that an application be implemented as a set of software entities that run on remote machines with distributed computations and decentralization of resources and knowledge. This section presents the implementation of this MAS model, which provides a tool to assist in the management of administrative work of the School; we let's call it "Smart Schooling".

The aim is to automate more school processes by integrating several autonomous entities (agents) and mobile agents that allow this model to be scalable, extensible and adaptable to the real architecture of the university.

This model allows us to reduce the cost of time and staff and facilitates the work of administrators. Figure 4 illustrates the different agents in our MAS model, as well as their containers. Each container represents the life environment of the agents it contains. The agents are distributed over different containers.

Container 4 houses the agents of the schools. In reality, each school agent will be deployed in a container on a remote host. All you need to do is add the IP address of the host of that container to enable communication between these containers. Simulation of the MAS model agents in this platform is necessary in order to verify the communication between these agents in the JADE graphical interface.

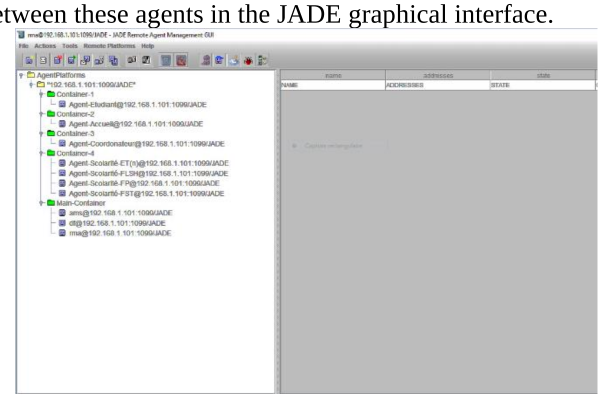

Fig. 4 MAS model agents and their containers

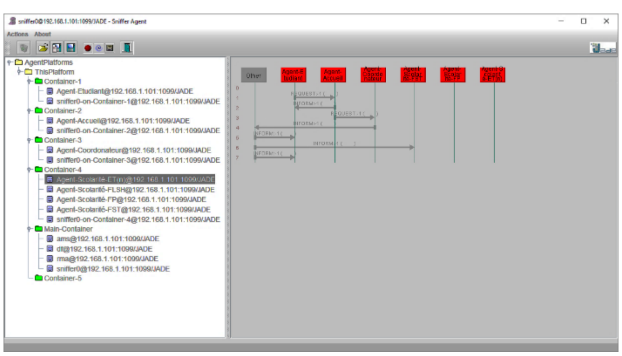

Fig. 5 Communication between SMA agents in JADE

5.2. Circulation of Information Between SMA Agents

Figure 5 illustrates the verification of message traffic between the agents on our platform. This communication is carried out using the sniffer agent, which makes it possible to the messages exchanged in the JADE agent platform. The two figures above describe the agents in our model and their distributed deployment environment, which is presented by the containers that container. In what follows, we will look at the steps and results of an attestation request experiment using our "Smart school" SMA model. This model is dedicated to requesting all types of scholarly attestation. We present the enrolment certificate as an experimental case.

5.3. Student Agent

The "Student-Agent" has an interface that allows the student to react with the MAS "Intelligent Schooling" model.

5.3.1. Initial Interface

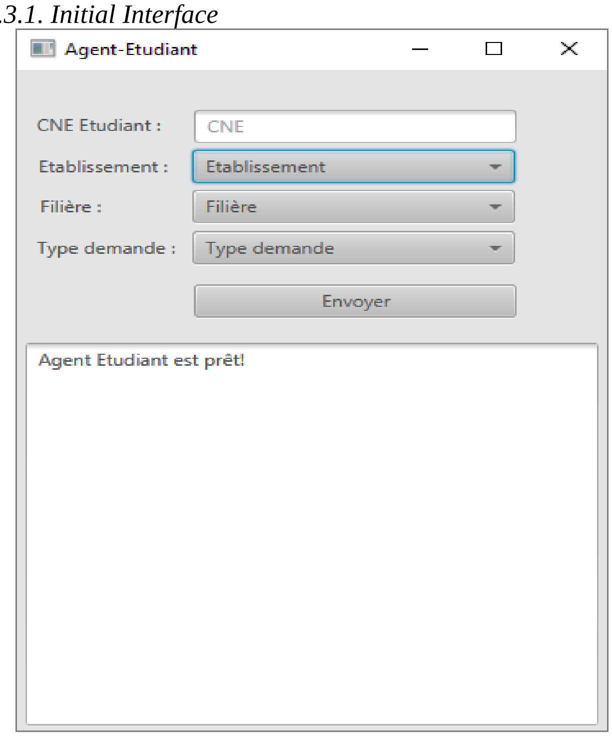

Fig. 6 Initial interface

The "Agent-Student" student agent has an interface that allows the student to interact with the MAS "Smart school" model. This interface provides the student with the following tasks:

- Enter the National Card Number (NCN)
- Select school, stream and type of request;
- Send request;
- Receive messages from other agents.

The figures 7 and 8 illustrate the "Student Agent" interface at different stages: This stage presents the "student agent" interface before the student initialises the parameters by the student.

5.3.2. Interface after Fields have been Filled in

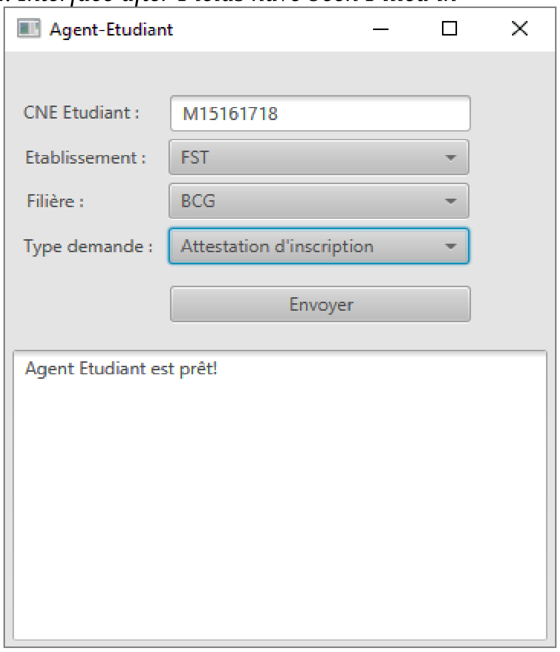

Fig. 7 Interface after filling in fields

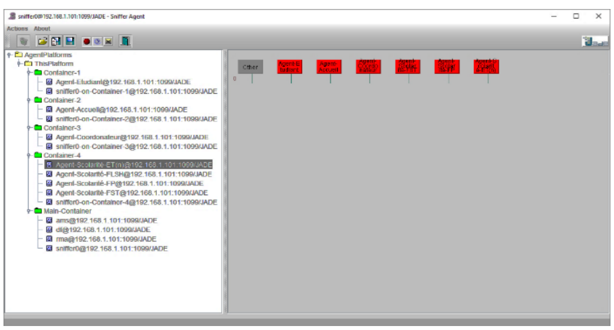

Fig. 8 Communication diagram before sending the request

This stage shows the "Student Agent" interface once all the fields required to request a certificate of enrollment have been filled in. To monitor communication between the agents during the application stages, we'll use JADE'S sniffer agent. Before sending the request, the communication diagram between agents is in the following form. The student sends his request by pressing the "Send" button. Received by the "Reception Agent".

5.4. Reception Agent

The "Agent-Accueil" reception agent's main role is to check the validity of the application received from the student agent. This check is carried out by connecting to the "DB" profile database, which contains the profiles of all the students who have sent requests.

This agent performs the following tasks:

- Check NCN validity;
- Connect to DB profile and check the validity of request;
- Send a message to the student agent in the negative case and stop the request process;
- Save the request in the DB profile in the positive case; Send a message to the student agent to inform him/her that the request has been registered;
- Send the request to the coordinating agent in the positive case.

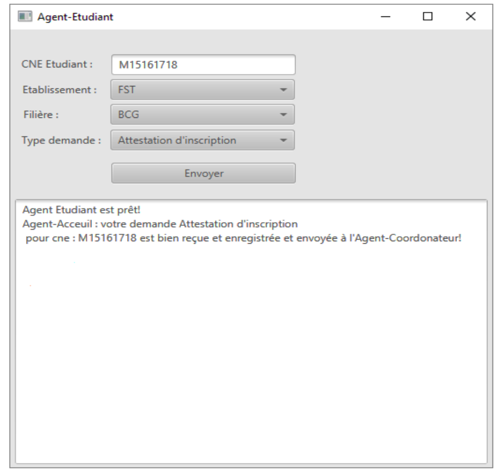

Fig. 9 Receive a message from reception agent

5.5. Coordinating Agent

This agent is responsible for identifying the establishment indicated in the application received from the reception agent.

- This agent performs the following tasks:
- Identify the destination establishment;
- Deploy a mobile agent and assign the request to it;
- Send the mobile agent to the schooling agent's container
- identify ;
- Send a status message to the student agent.

For each request received by the coordinating agent, a mobile agent will be deployed to move to the container of the schooling agent concerned by the request. Upon container, the mobile agent registers the request in the database, sends the request to the schooling agent on that container, and finally sends a status message to the student agent. Then, this agent dies.Figure 10 illustrates the status of our request sent by the student agent. In this stage, the appearance of a mobile agent is deployed by the coordinating agent for this request. This mobile agent, hosted in container 5, will move towards the student agent.

5.5.1. Schooling Agent

This agent owns a main graphical interface, which presents a dashboard of these functionalities. It is responsible for managing requests. The main tasks carried out by this agent are as follows:

- Receive requests from the mobile agent;
- Connect to the application database;
- Process the requests;
- Send a status message to the student agent;
- Print certificates.

Figure 11 shows the results once the request has reached the level of the schooling agent.

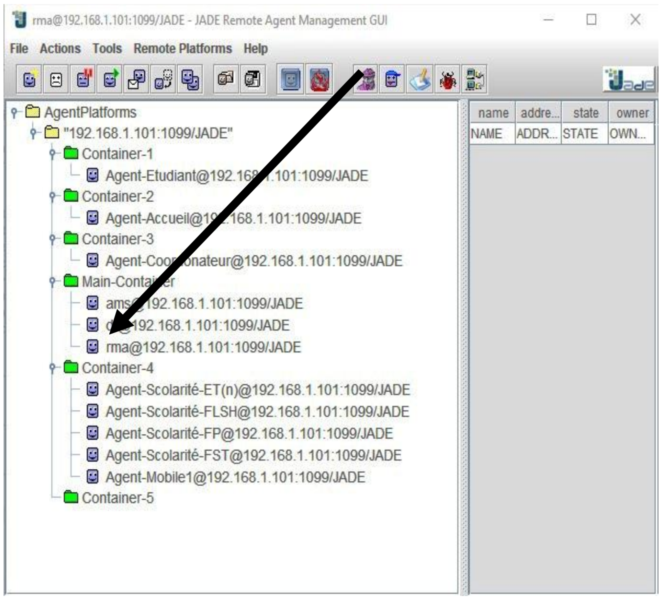

Fig. 10 Deployment of the mobile agent

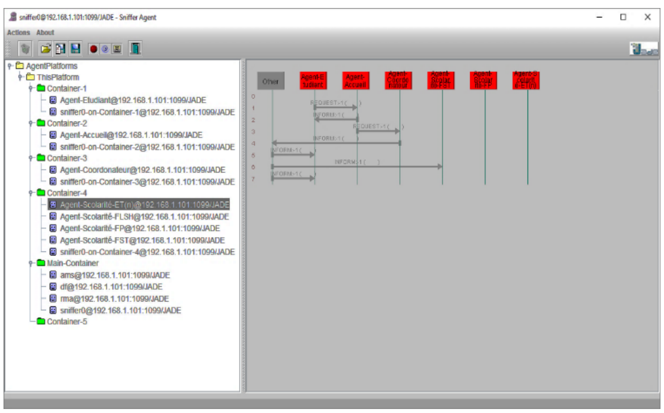
Fig. 11 Interaction diagram between the agents in our request

The "other" agent plays the role of the mobile agent in Figure 12.

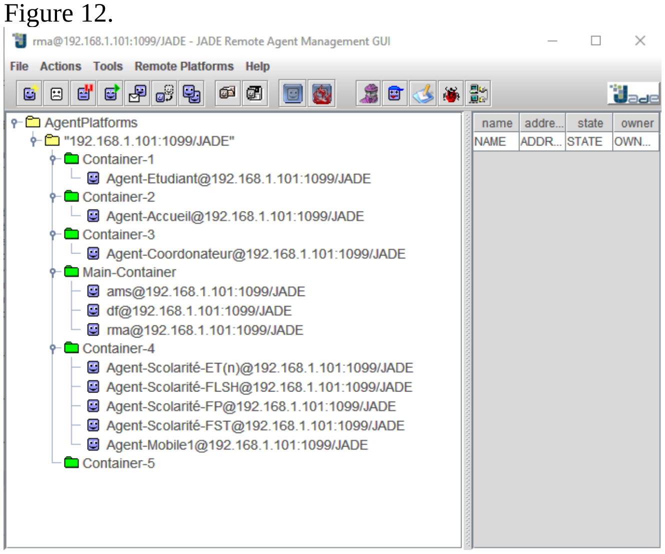

Fig. 12 Mobile agent has left container 5

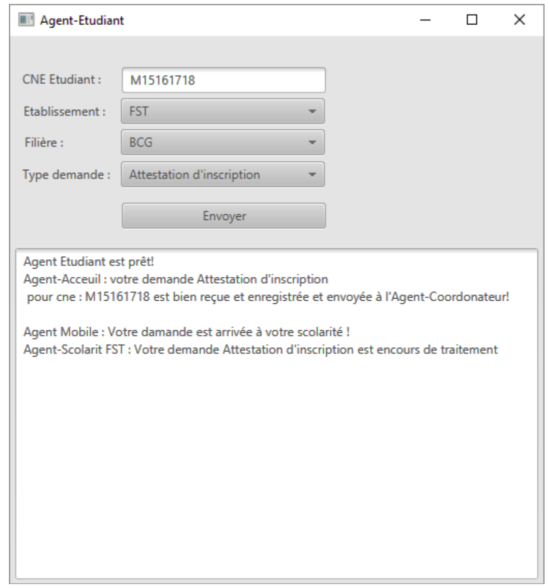

Fig. 13 Messages received by the student agent at this stage

The mobile agent died after sending the message, which appears in Figure 13. Figure 14 shows the Faculty of Science and Technology (FST) schooling agent, an agent with a graphical interface. Our request has arrived at this agent, and its status is "Processing in progress". The FST schooling Agent connects to his database, processes the request and finally prints it out. Figure 16 illustrates the selection of our request and its printing by the FST schooling agent. The FST schooling Agent clicks on the "Update" Botton, and the status of the request changes to "Processed".

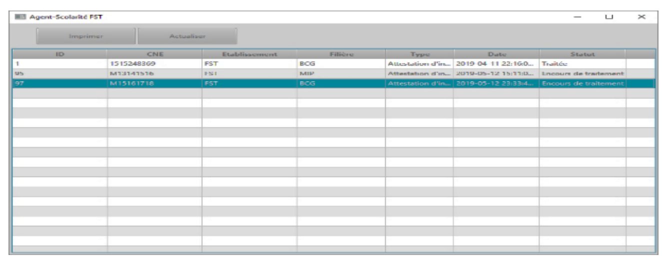
Fig. 14 Faculty of Science and Technology school agent interface

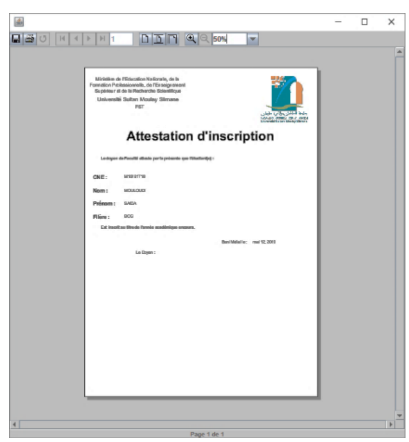

Fig. 15 Printed attestation

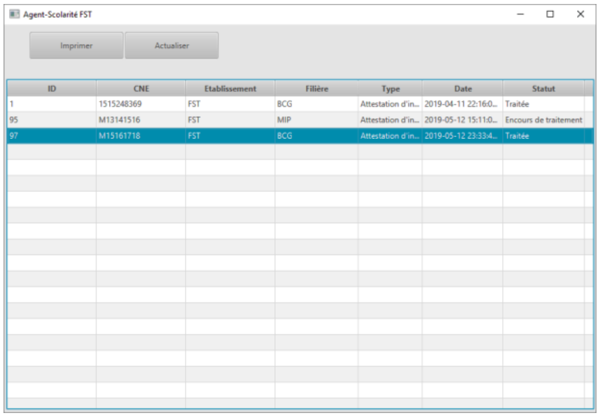

Fig. 16 Request status after processing

The student receives the last message in the following figure 17.

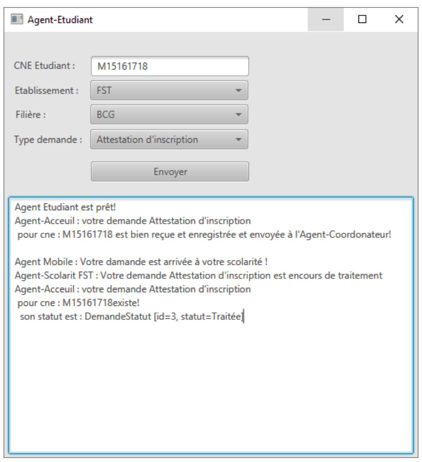

Fig. 17 Messages received by the student in all stages

5.6. A Second Request (Multidisciplinary Faculty)

In this section, we will present the steps involved in another application made by the polydisciplinary faculty after the first FST application, but only the first and last steps will be presented here; the other steps are similar to those of the first application.

5.6.1. Initial step: "Student Agent" Request Preparation

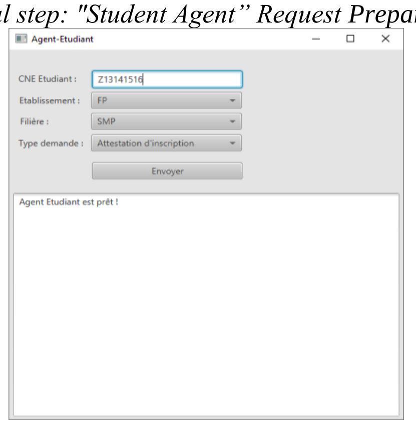

Fig. 18 Filling in request fields

At this point, the interaction simulation diagram, Figure 19, is still in its state after the first FST request.

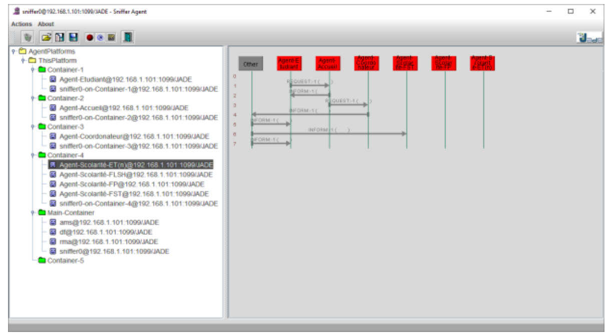

Fig. 19 Initial state of the diagram for this second request

5.6.2. Final Step: The Request has Reached the "Multidisciplinary Faculty Schooling Agent"

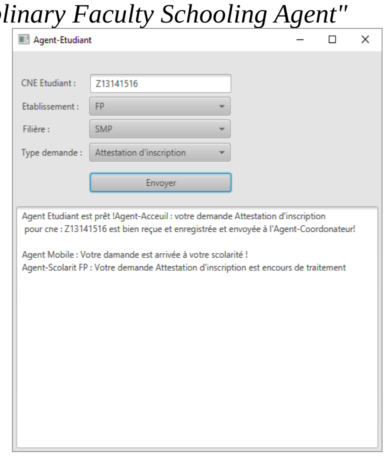

Fig. 20 Sent messages

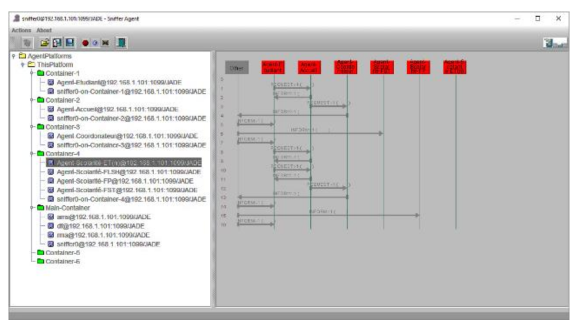

Fig. 21 Interaction diagram after the second request

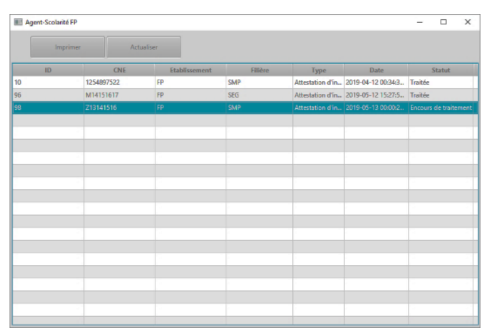

Fig. 22 Multidisciplinary faculty schooling agent

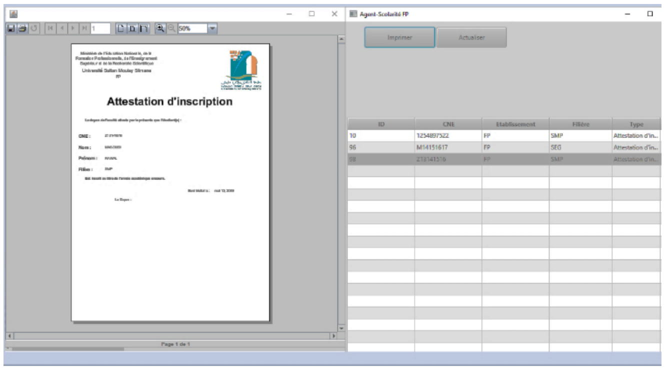

Fig. 23 Printed attestation

The schooling agent's interface can be upgraded by adding other functionalities. His interface thus presents an evolving dashboard without affecting the other agents. To help the schooling agent perform other tasks it needs to perform, such as statistics or others, we can implement other agents separately, assign them tasks and connect them to the graphical interface via Bottons, which launches their deployments without changing the architecture of our MAS model. This scalability is a benefit of the modularity property of Multiagent Systems.

6. Conclusion

The explosion of Internet and network technology has helped to shake up many of the habits established decades ago. Paper documents exchanged from hand to hand are gradually giving way to electronic documents transmitted automatically by machines. As a result, the fields of application for IT systems are becoming increasingly vast and complex, and they are being asked to manage ever greater interactivity, responsiveness and mobility in their most widespread uses. Thus, the emergence of multiagent systems (MAS) brings a new dimension to the concept of modeling to represent a real-world application with an appropriate degree of complexity and dynamics. From this work, we have concluded that the development of object-based distributed applications requires more time and essential tools for the realization of these applications, which remain limited in adaptation and scalability. Despite these difficulties in the design and realization of MAS models, they have major advantages linked to their flexibility and dynamism in distribution. As a way of improving this work in the future, we propose the study and realization of an approach that combines agents and software components to achieve more efficient and less costly results.

References

[1] Zhang Xue-ling, and MA Chaoqun, "The Employment of Target Administration Theory in the Administration of Teaching Staff in Colleges and Universities," *Journal of Xianyang Teachers College*, no. 6, pp. 94-96, 2005. [Publisher Link]

[2] Robert Allen, Rémi Douence, and David Garlan, "Specifying and Analyzing Dynamic Software Architectures," *Fundamental Approaches to Software Engineering, Lecture Notes in Computer Science*, pp. 21-37, 1998. [CrossRef] [Google Scholar] [Publisher Link]

[3] Michael Wooldridge, *An Introduction to Multi Agent Systems*, John Wiley & Sons, pp. 1-461, 2009. [Google Scholar] [Publisher Link]

[4] A. Halberstadt, and L. Mui, "Group and Reputation Modeling in Multi-Agent Systems," *Proceeding Goddard/JPL Workshop on Radical Agents Concepts, NASA Goddard Space Flight Center*, 2001. [Google Scholar]

[5] Thierry Bouron, Jacques Ferber, and Fabien Samuel, "MAGES. A Multi-Agent Testbed for Heterogeneous Agents," *Decentralized Artificial Intelligence*, vol. 2, pp. 195-214, 1991. [Google Scholar]

[6] Alan H. Bond, and Les Gasser, *Reading in Distributed Artificial Intelligence*, Elsevier Science, pp. 1-668, 1988. [Google Scholar] [Publisher Link]

[7] Victor R. Lesser, and Daniel G. Corkill, "The Distributed Vehicle Monitoring Testbed: A Tool for Investigating Distributed Problem Solving Network," *AI Magazine*, vol. 4, no. 3, pp. 15-33, 1983. [CrossRef] [Google Scholar] [Publisher Link]

[8] Les Gasser, "Social Conceptions of Knowledge and Action: DAI Foundations and Open Systems Semantics," *Artificial Intelligence*, vol. 47, no. 1-3, pp. 107-138, 1991. [CrossRef] [Google Scholar] [Publisher Link]

[9] C. Baeijs, Y. Demazeau, "Organizations in Multi-Agent Systems," *4th Days of the GDR PRC IA on Multi-Agent Systems*, 1996. [Google Scholar]

[10] Jaime Simão Sichman, and Rosaria Conte, "On Personal and Role Mental Attitudes: A Preliminary Dependence-Based Analysis," *Advances in Artificial Intelligence, Brazilian Symposium on Artificial Intelligence*, pp. 1-10, 1998. [CrossRef] [Google Scholar] [Publisher Link]

[11] Qingtan Meng, Qian Ma, and Guopeng Zhou, "Distributed Optimization for Nonlinear Multi-Agent Systems with an Upper Triangular Structure" *Information Sciences*, vol. 641, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[12] Zalimhan Nagoev et al., "Multi-Agent Neurocognitive Architecture of an Intelligent Agent Pattern Recognition System" *Procedia Computer Science*, vol. 213, pp. 504-509, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[13] Leila Amgoud, Nicolas Maudet, and Simon Parsons, *An Argumentation-Based Semantics for Agent Communication Languages*, 15th European Conference on Artificial Intelligence, pp. 38-42, 2002. [Google Scholar] [Publisher Link]

[14] Leila Amgoud, and Simon Parsons, "Agent Dialogues with Conflicting Preferences," *Intelligent Agents VIII, ATAL 2001, Lecture Notes in Computer Science*, pp. 190-205, 2002. [CrossRef] [Google Scholar] [Publisher Link]

[15] Scott A. De Loach, and Mark Wood, "Developing Multiagent Systems with Agent Tool," *Intelligent Agents VII Agent Theories Architectures and Languages*, *ATAL 2000*, *Lecture Notes in Computer Science*, pp. 46-60, 2001. [CrossRef] [Google Scholar] [Publisher Link]
[16] Giovanni Caire, and David Cabanillas, "JADE Tutorial: Creating and Using Application Specific Ontologies," *Java Agent Development Framework*, pp. 1-30, 2004. [Google Scholar]
[17] "*FIPA ACL Communicative Act Library Specification*," Foundation for Intelligent Physical Agents, Techenical Report, 2002. [Google Scholar] [Publisher Link]
[18] Jacques Ferber, *Multi-Agent Systems towards Collective Intelligence*, Inter Editions, pp. 1-522, 1995. [Google Scholar] [Publisher Link]
[19] Olivier Boissier, Sylvain Gitton, and Pierre Glize, *Characteristics of Systems and Applications*, Multi-agent Systems/French Observatory of Advanced Techniques, pp. 1-25, 2004. [Google Scholar] [Publisher Link]
[20] "*FIPA ACL Message Structure Specification*," Foundation for Intelligent Physical Agents, Techenical Report, 2002. [Google Scholar] [Publisher Link]
[21] Christos G. Cassandras, Xu Chu Ding, and Xuchao Lin, "An Optimal Control Approach for the Persistent Monitoring Problem," *2011 50th IEEE Conference on Decision and Control and European Control Conference*, Orlando, FL, USA, pp. 2907-2912, 2011. [CrossRef] [Google Scholar] [Publisher Link]
[22] Morris C. Berenbaum, "Criteria for Analyzing Interactions between Biologically Active Agents," *Advances in Cancer Research*, vol. 35, pp. 269-335, 1981. [CrossRef] [Google Scholar] [Publisher Link]