ISSN 2347 - 3983
Volume 8. No. 5, May 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter105852020.pdf
https://doi.org/10.30534/ijeter/2020/105852020

Multi-Agent System Architecture Oriented Prometheus Methodology Design for Multi-modal Transportation

Jihane LARIOUI [1], **Abdeltif EL BYED** [2]
Laboratory of computer science and modelling of decision support systems
HASSAN 2 University Faculty of Science Ain Chock
Casablanca, Morocco
[1] jihane.larioui-etu@etu.univh2c.ma
[2] abdeltif.elbyed@univh2c.ma

ABSTRACT

Urban mobility is the subject of several current studies and attracts more and more researchers and industrialists. This subject has often been linked to transportation problems, particularly in terms of travel difficulties. In addition, the design of the multimodal network can be difficult to implement and decision-making is one of the major challenges of this network. The complexity of traveling in a multimodal network has necessitated the development of an intelligent solution that will facilitate decision-making and route planning. The main objective of this document is to design a multi-agent architecture using PROMETHEUS method to effectively manage the movements of travelers in a multimodal network. The proposed MAS architecture includes six types of agents: Personal Travel Agent, Information Agent, Directory Selecting Agent, Sorting Agent, Calculating Agent and Decision-Making Agent. Each Agent plays a special role in the system and acts respectively during each route planning step.

Key words : Urban Mobility; multi-agent system (MAS); Intelligent transport system (ITS); Prometheus Methodology; Route Planning; multi-modal Itinerary; Prometheus Design Tool.

1. INTRODUCTION

The urban mobility is a concept in constant evolution: the number of citizens has multiplied by 15 during the century and the size of cities has greatly increased. As a result, the city has become a complex system that requires a lot of expertise in different areas of research. This concept plays an important role in large cities. Mobility is increasing rapidly in different regions of the world due to population growth and technological developments. Where necessary, effective methods are needed to manage traffic so that its side effects such as traffic congestion, accidents and injuries, wasted resources, noise / air pollution can be minimized. So as to reduce traffic problems, various strategies have been proposed, notably the improvement of public transport systems and the development of intelligent transport systems (ITS). Multi-agent systems (SMA) have shown their relevance for the design of complex and robust distributed applications (logically or physically). The concept of agent is today more than an effective technology, it represents a new paradigm for the development of software in which the agent is an autonomous software, which has a goal, evolves in an environment and interacts with others agents using languages and protocols. Often the agent is viewed as an "intelligent" object or as a level of abstraction above objects and components. Object-oriented development methods are not directly applicable to SMA development. It then became necessary to extend or develop new models, new methodologies and new tools adapted to the concept of agent. The information system presented is based on a multi-agent architecture to associate user requests with information linked to the different transport operators. The system allows choosing the modes of transport to combine and offering itineraries that meet itinerary requests. Thus, the traveler will no longer have to consult several transport sites in order to plan his trip [1,2] because he can express his preferences between different modes of transport and define a decreasing order of priority with several criteria such as time, number matching, cost and safety.
To deal with this situation, decision-making requires an intelligent and efficient modeling methodology to support the main tasks of urban mobility. Consequently, the multi-agent system used breaks down complex problems into small sub-problems that are easy to manage and solve by the individual agent in cooperation. However, one of the major challenges in the field of artificial intelligence is designing agents who will be able to work together toward the same goal [5]. In fact, to develop this system, the authors chose the Prometheus methodology since it is generally conceived as a complete, practical and easily achievable methodology specifically for the design of multi-agent systems, providing everything necessary for the specification and design of agents [4].
The rest of this article is organized as follows: Section 2 begins with a Background about ITS in urban mobility and

discusses the relevance of Prometheus methodology in such systems. Section 3 presents The Prometheus Methodology and tools support. Section 4 discusses the Prometheus methodology and presents the design of the proposed agent system. The final section concludes the paper with some remarks and perspectives.

2. BACKGROUND

Nowadays, urban mobility is the subject of several studies and attracts more and more researchers. Several works have been interested in solving this problem via the paradigm of multi-agent information systems. The latter presents an effective way to solve the problem of multimodal passenger information since it allows personalization, collection, integration and processing of data.

Usually, most research contributions consider a multi-agent system to improve route planning in a multimodal network, other than that, they have not considered using a design methodology to support the main tasks of the multimodal network.

In the literature, agent-based information systems are commonly referred to as Advanced Traveler Information Systems (ATIS). Indeed, ATIS offers to travelers a real-time assistance and traffic information, including trip schedules and navigation. It presents information about congestion, weather conditions and accidents [18, 19]. In fact, Multi-agent systems technology (MAS) is suitable for the development of complex and distributed systems. In this context, several methods have been proposed in order to meet the demand for agent-oriented software, in particular, Tropos, Gaia, MaSE, INGENIAS and Prometheus.

Actually, using model-driven engineering techniques (MDE) throughout the software development process is increasing interest of researchers [15]. The key idea behind this paradigm is that if a model drives the development, there will be significant benefits in fundamental aspects such as productivity, portability, interoperability and maintenance. Thus, in the MAS paradigm, it is better to opt for a methodology such as Prometheus, which supports this approach. Nevertheless, a perfect methodology that satisfies all of the developer's needs cannot be found. This is the reason why different methodologies are studied to create a new one. In fact, in the literature, there are methodologies that are influenced by other ones, which have already been proposed previously. However, any methodology among those cited is able to handle all the specifications of MAS, as defined by [10].

In this article, we consider the Prometheus Methodology since it includes all the stages from specification of requirements to implementation and deployment. According to [6, 8] Prometheus is a methodology for the development of intelligent agents that differs from the others because it is detailed and complete, covering all the tasks necessary for the development of intelligent agent systems.

We could have opted for the INGENIAS method for example, but in our opinion, the process followed in this method during the analysis and design phases of the MAS is very complex and difficult to understand, because it is not clear how the different models are built during the phases, despite the general guidelines documented. In addition, INGENIAS does not provide any mechanism to discover who will be the agents of the system and their interactions and does not take into account the development platform at the detailed design phase [17].

Otherwise, a design tool for Prometheus, called Prometheus Design Tool (PDT), is available and allows users to create and modify projects developed with the Prometheus methodology [14]. This tool ensures that there are not some inconsistencies, such as the use of wrong diagram notation, and allows exporting individual diagrams and generating a report with the entire project.

In this article, we present an information system in the context of urban mobility based on a semantic approach. This system covers all modes of transport and is capable of managing route planning as well as answering questions related to transport costs, travel time, route, number of mode changes and safety [3, 4]. To our knowledge, none of the existing agent-based work has been sufficiently demonstrated to support the tasks of a multimodal transport system.

However, the main scientific contribution of this article lies in designing the architecture of the multi-agent system under the Prometheus methodology by using the Prometheus design tool (PDT) to identify the types of agent and the architecture of each one of them.

3. PROMETHEUS METHOLOGY AND TOOLS

3.1 PROMETHEUS Methodology Overview

The Prometheus method, from Padgham and Winikoff, is a Belief – Desire – Intention (BDI) agent development method, whose JACK platform represents the execution environment [7]. Thus, the choices of modeling and specification are directly influenced by the technical constraints imposed by Jack. Based on the object paradigm, Prometheus takes into account modularity and complexity management thanks to the decomposition of agents into activities and activities into sub-activities. Since Prometheus is coupled with Jack, the Java language is preferable for implementation.

In particular, the Prometheus methodology as described in [6] contains a description of concepts for designing agents, a process, a number of notations for capturing designs, as well as many tips or techniques that give advice on how to carry out the steps of Prometheus process.

The Prometheus methodology is an agent oriented software engineering methodology [8]. It consists of the following three phases:

- ***The system specification phase*** that focuses on identifying the goals and basic functionalities of the system, along with inputs (precepts) and outputs (actions)

- ***The architectural design phase*** that uses the outputs from the previous phase to determine which agent types the system will contain and how they will interact
- ***The detailed design phase*** that looks at the internals of each agent and how it will accomplish its task within the overall system.

The following sections summarize the results of the different design phases.
Figure 1 indicates the main design artifacts arising from each of these phases as well as some of the intermediary items and relationships between them.

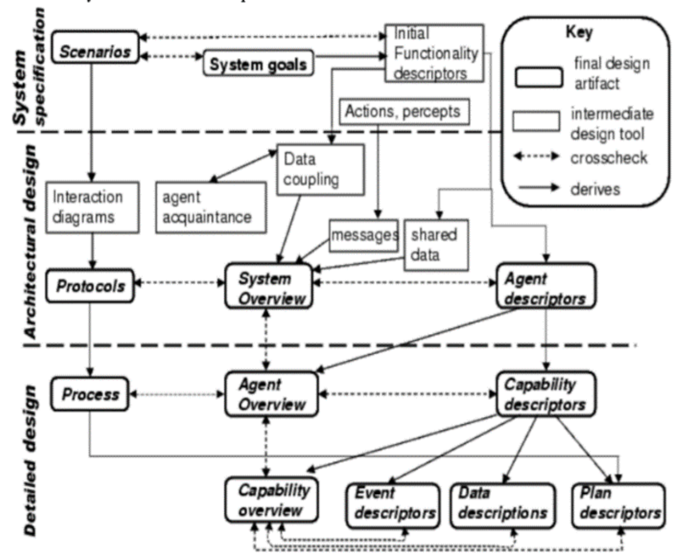

Figure 1: Prometheus Methodology Phases

Thus, the choice of the Prometheus methodology is maintained by the fact that this methodology is complete, practical and easily implementable specifically for the design of multi-agent systems, providing everything necessary to specify and design the agents [8].

3.2 Tools Support

Among the practical aspects that distinguish the Prometheus methodology from other methods is the fact that there is a free design tool available called Prometheus Design Tool (PDT). Prometheus Design Tool (PDT) is a graphic editor that covers all the design tasks specified in the Prometheus methodology for the design of agent systems [14].
Like most modern software engineering methodologies, Prometheus is responsible for ensuring inconsistencies on all changes made to ensure that the design remains consistent and unified. The program is written in Java and its main features include structured text descriptors, the propagation of information from one part of the design to another, consistency checking, hierarchical views and generation of reports. Detailed output diagrams from the design phase can be easily transformed into JACK agent code. A latest version of PDT is now integrated into the Eclipse platform, allowing users to achieve the entire development process of an agent-based application in a single IDE and inherit a set of features of product development that Eclipse offers [13].
Beside of that, there is another tool that meets the design need using Prometheus methodology, it is the JACK development environment (JDE). The latter offers a design tool for drawing Prometheus style presentation diagrams. The JDE can then generate skeleton code from these diagrams. Indeed, it is a framework in Java that ensures that the modifications made to the code are reflected in the design drawings [7].

4. DESIGN BASED PROMETHEUS METHODOLOGY

4.1. System Specification

Designing a multi-agent system is a complicated process. In fact, such systems contain hundreds of agents running in real time, perceiving, communicating, negotiating, making decisions and executing. Such a complex system must be designed properly so that it can include not only all the functionality necessary to keep the system performing, but also to ensure its evolution [9]. In order to remedy this effect, we use the Prometheus methodology since it is complete and detailed. The main purpose of this methodology is to guide through the decomposition process. Indeed, it is a question of breaking down each complex objective into several sub-objectives that are smaller and more manageable. In this way, the system will be more understandable and therefore easier to design [16, 17].
The system specification represents the first phase of the Prometheus methodology. In this section, we start with a brief description of what our proposed system should be able to do. Next, we detail the system goals and then we describe a set of system scenarios.

A. System Description

Previous works for the development of an advanced and intelligent multimodal information system has been proposed to develop an architecture based on multi-agent systems [1, 2]. We propose a new approach that combines between semantic web technologies and the multi-agent paradigm to design an advanced multi-agent information system for multimodal transportation [3]. It would be an effective way to communicate and coordinate among the different agents and improve the decision-making process.
The multi-agent system to be developed should be in charge of all the main tasks of urban mobility. Indeed, this system should cover all modes of transport and manage route planning according to user preferences. In addition, this system should access the database of the various transport operators and integrate the results generated by the various agents that compose it.
The agents that typeset our architecture include six types of agents: *Personal Travel Agent (**PTA**), Information Agent (**IA**), Directory Selecting Agent (**DSA**), Sorting Agent (**SA**), Calculating Agent (**CA**) and Decision-Making Agent (**DMA**).*
Each Agent plays a special role in the system and acts respectively during each step of the route planning. Figure 2 shows the detailed design of the proposed multi-agent system architecture.

First, the user sends his request, specifies his departure and arrival stations and sets his preferences in descending order of priority with several criteria such as time, number of connections, cost and safety. The PTA agent receives the request and divides it into two sub-requests: the first sub-request concerns only the departure and arrival of the user, it is sent directly to the DSA agent while for the second concerns preferences, which is sent to the SA agent. As soon as the DSA agent receives a request, he questions the IA agents who will be required to provide the necessary information in order to identify the area of research. Thus, the DSA agent determines the different route suggestions between departure and arrival by identifying the IA agents connecting the departure and arrival points who will have to collaborate together to determine the final route. The DSA agent then transmits all of the suggestions to the SA agent. The latter bases on these routes to calculate the final route according to user preferences. The CA agent calculates the parameters necessary for decision making for each proposed route and transmits them to the DMA agent. The latter is based on the TOPSIS method "*Technique for order of preference by similarity to the ideal solution*" as MCDM methodology to facilitate decision-making and choose the route that will satisfy the user's preferences. The DMA agent then sends the final response (the chosen route) to the SA agent who will send the information to the PTA agent to answer to user.

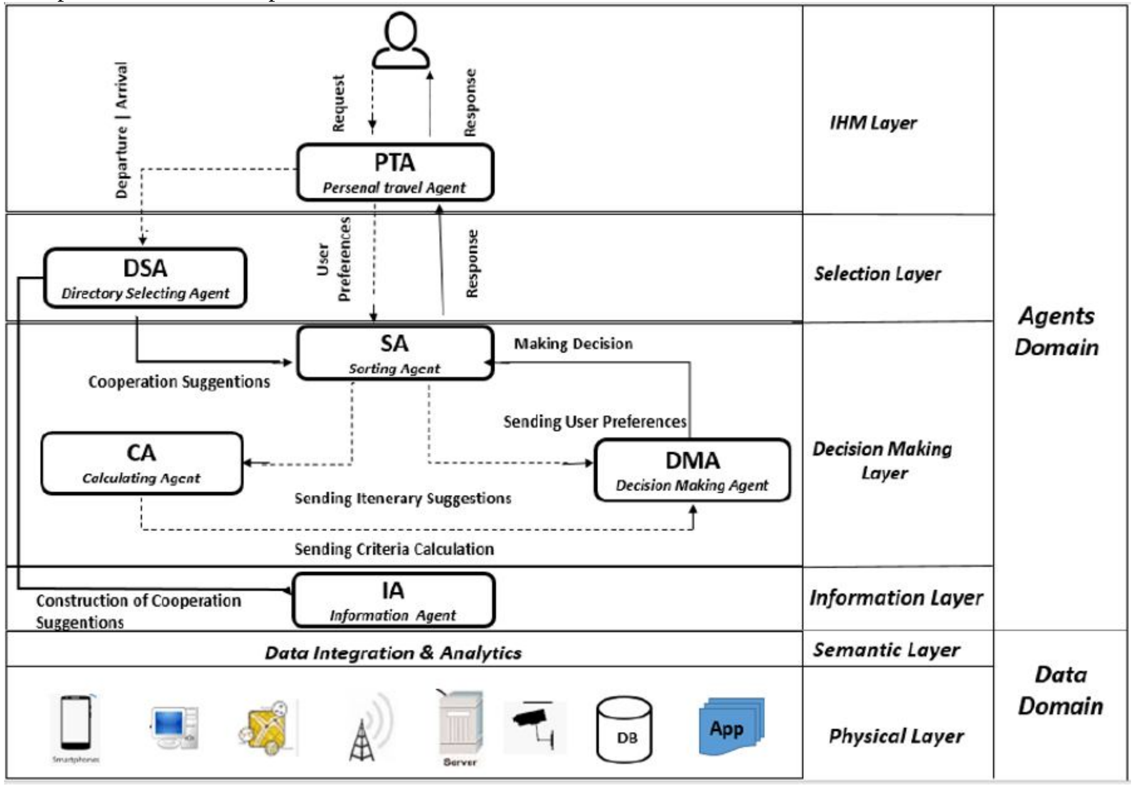

Figure 2: Multi-Agent Information System Architecture

B. *System Goals*

As discussed in the previous section, we conclude that the main tasks of the proposed system are as follow:

- ***Assistance of passengers (Keep the passenger informed of his journey in real time)***
- ***Data Collection and Integration***
- ***Analysis of preference criteria***
- ***Decision making***
- ***Route Planning***

Thus, the main goals of our system are represented as follow:

- **G1 Assistance Process:** The agent system is responsible for accompanying and assisting the user by obtaining the user's inputs and presenting the information.
- **G2 Data Collection Process:** The agent system should access to all information on the network, collect and integrate the data in the system.
- **G3 Identification Process:** The agent system has to collect data based on the departure and arrival request and list all the suggestion itineraries.
- **G4 Sorting Process:** The agent system examines and analyses the criteria of preferences expressed by the user.
- **G5 Decision-making Process:** The agent system makes decision and finds the optimal itinerary.
- **G6 Calculation Process:** The agent system calculates the final route that meets the needs of the user.

The main goals of the agent system are briefly presented in the Figure 3 below

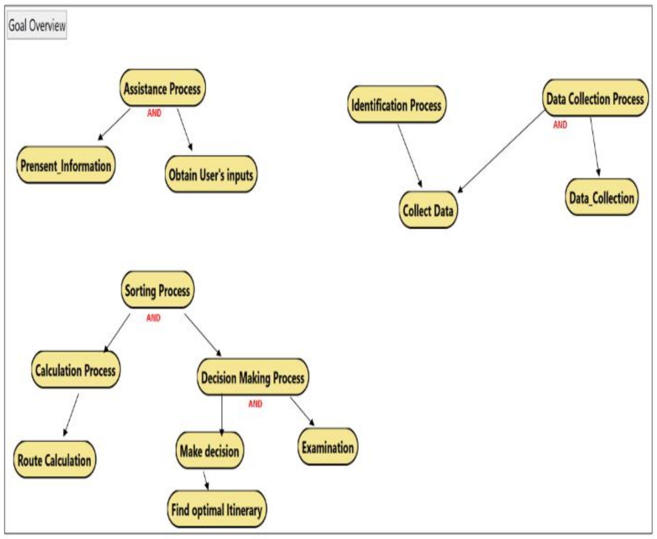

Figure 3: Goal Overview Diagram

C. System Scenarios

In this part, we detail the scenarios that may occur in the system and relate all the execution steps. The scenarios bellow show the normal operation of the system.

- ***S1: The Assistance Process Scenario:*** Once the user sends the requests, the agent system split the request into two sub-request and send them to Scenario S2 and S3.
- ***S2: The Identification Process Scenario:*** Once the points of departure and arrival are received, the agent system has to defines the search domain by specifying the information agents that will work together to plan the route and propose the different combinations of possible information agents. Then, sends a list of suggestions to Scenario S4.
- ***S3: Data Collection process Scenario:*** The agent system searches, collects and integrates Data regarding the departure and arrival requested and send the response to Scenario S2.
- ***S4: The Sorting Process Scenario:*** After receiving the user's preferences and the list of suggestions, the agent system examines the different routes and decides how to treat them according to the preferences of the users. Scenario S5 and S6.
- **S5: The Calculation Process Scenario:** After getting the list of suggestions itineraries, the agent system calculates for each route its necessary parameters. These parameters are calculated based on user preferences (travel time, number of mode changes, cost and safety) and send the results to Scenario S6.
- **S6: The Decision Making Process Scenario:** Once the calculated parameters are defined, the agent system defines the optimal itinerary using a MCDM methodology that will satisfy the user's preferences.

Figure 4 illustrates the scenarios and the connections between them.

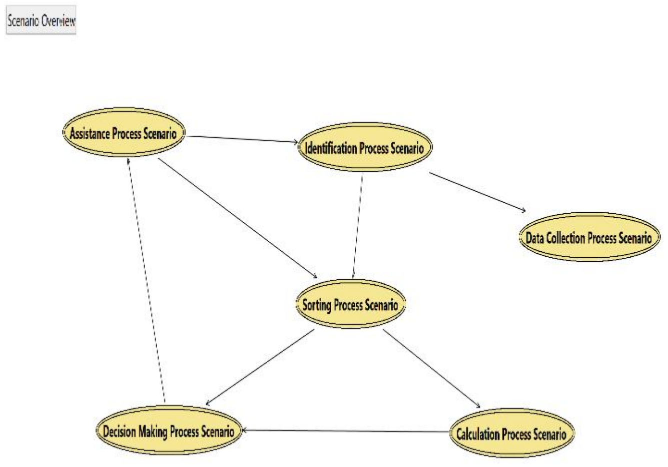
Scenario Overview
Assistance Process Scenario
Identification Process Scenario
Data Collection Process Scenario
Sorting Process Scenario
Decision Making Process Scenario
Calculation Process Scenario

Figure 4: System Scenario Overview Diagram

4.2. Architectural Design

The purpose of this section is to define what agents are to be parts of the system and how they interact with each other and which external connections are required. The artifacts produces during the system specification are used as a basis for developing the high-level design of agent system. This section presents the architecture of the multi-agent system. It starts with a detailed description of how the agents are distributed in the environment and continues with an illustration of the overall interaction between them. Figure 5 presents the Agent Role Grouping Overview.

- ***The PTA "Personal Travel Agent"*** representing the Human Machine Interface. **R1: Online Interaction**
- ***The DSA "Directory Selecting Agent",*** which defines the search domain by specifying the information agents that will work together to plan the route and suggest different possible itineraries. **R2: Make Suggestion**
- ***The SA "Sorting Agent"***, which examines the different routes proposed by the DSA and decides how to treat them according to the preferences of the users. **R3: Sorting**
- ***The CA "Calculating Agent"***, which takes up the routes proposed by the DSA in order to calculate for each route its necessary parameters. These parameters are calculated based on user preferences (travel time, number of mode changes, cost and safety). **R4: Itinerary Calculation**
- ***The DMA "Decision Making Agent"*** which makes decision and choose the itinerary that will satisfy the user's preferences. **R5: Make Decision**
- ***The IA "Information Agents",*** which are responsible for searching, collecting, integrating and manipulating the information from the different sources of information. **R6: Data Collection**

The PTA agent sends messages to Sorting agent and Directory selecting agent. The Directory Selecting agent sends requests to the Information agent and wait for the response. The sorting agent communicates in the some way and in parallel with the calculation agent and Decision making agent. The agent types of this architecture are defined by considering the roles and scenario of the system specification.

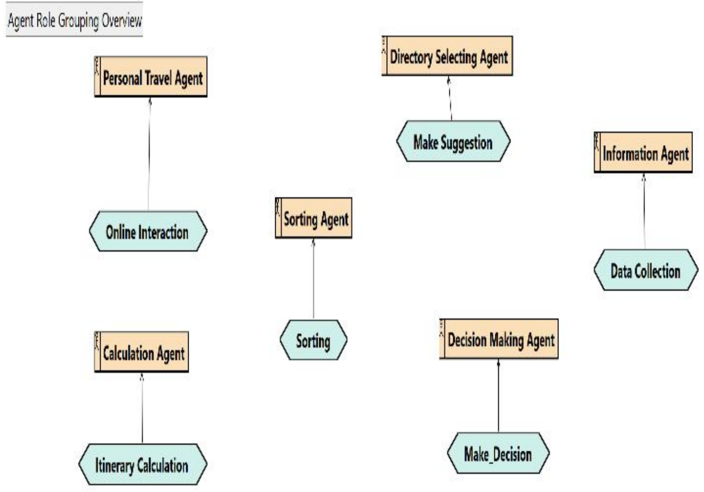

Figure 5: Agent Role Grouping Overview

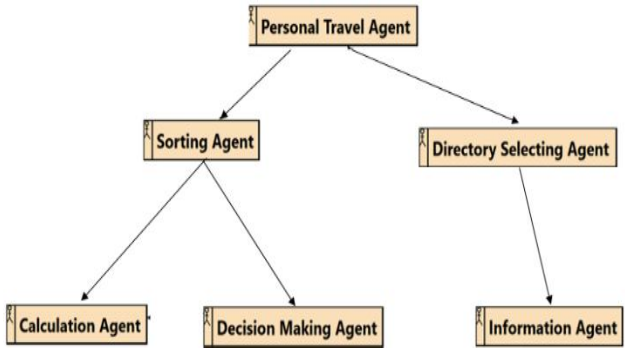

Figure 6: Agent Acquaintance Diagram

The interaction between agents capture the dynamic aspects of the system, Fig. 6 shows an acquaintance diagram that illustrates how the agents are connected.

4.3. Detailed Design

The final design phase builds upon the agent descriptors defined in the previous phase and further specifies the behaviour of every agent. This includes all triggers the agent responds to, how it responds accordingly, which agents it interacts with and how the agents communicate.

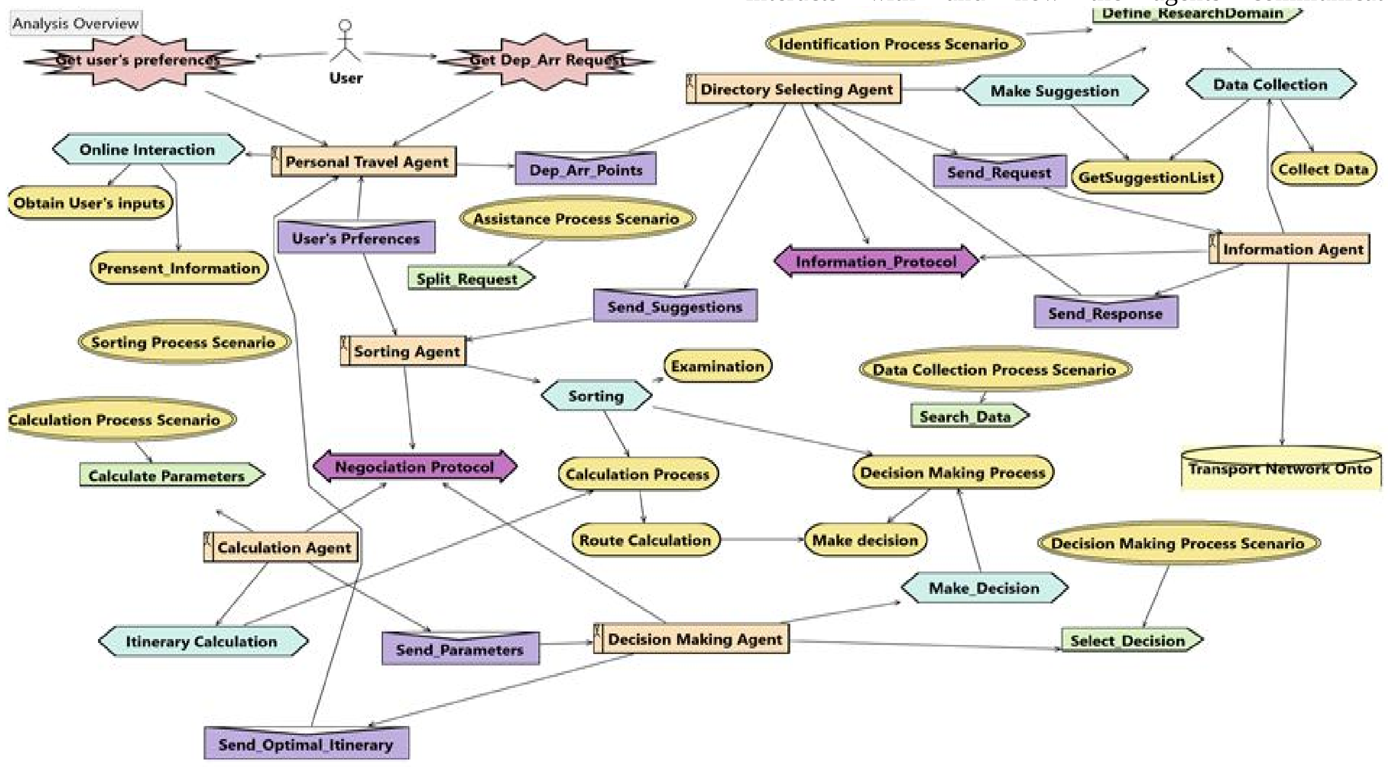

Figure 7: System Overview Diagram

The roles and tasks of these agents are described, and the overall interaction between them. Figure 7 provides an overview of the architecture with the main entities involved. The system overview diagram developed using PDT detailed design process. PDT can validate each diagram dynamically meanwhile the development process is running. The Figure 8 and 9 describes the Protocols of how the agents interacts and communicates to with each other.

The Figure 8 describes the Information Protocol AUML Diagram. This Information Protocol aims to ensure the communication between the Directory Selecting Agent DSA and the Information Agent IA from one hand, and between de Directory Selecting Agent DSA and the Sorting Agent SA on the other hand. First, the DSA sends the request to IA to have information in order to make a list of the suggestion itineraries. Then, the DSA send the suggestions to the SA.

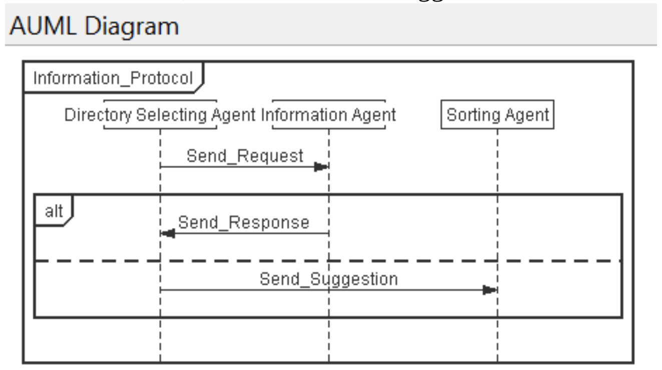

Figure 8. Information Protocol AUML Diagram

The Figure 9 presents the Negotiation Protocol AUML Diagram. This Negotiation Protocol guarantees the communication between the Sorting Agent SA, Calculation Agent CA and Decision-making Agent DMA. In fact, once the SA receives the suggestions, he sends them to the CA in order to calculate the parameters. Then, the CA sends those parameters to the DMA to help him to make decision and find the optimal itinerary.

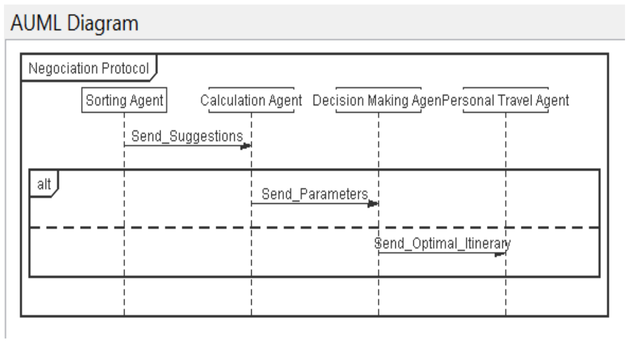

Figure 9. Negotiation Protocol AUML Diagram

5. CONCLUSION

This paper proposes an architecture for representing an advanced information support system for multimodal transportation network based on multi-agent systems. The multi-agent architecture proposed is efficient, flexible, designed to easily be adapted and to manage disturbances in the transport network. In this context, the design of the main architecture was described using Prometheus Methodology within PDT (Prometheus Design tools). The next step is to integrate a model of MCDM to improve the agents' decision-making, and to develop the calculation of the multi-modal itinerary of this system.

REFERENCES

1. J. LARIOUI, A. EL BYED, **A Multi-Agent Information System Architecture For Multimodal Transportation**, Embedded Systems and Artificial Intelligence Proceedings of ESAI 2019, Fez, Morocco, Pages 795-803 https://doi.org/10.1007/978-981-15-0947-6_75
2. J. LARIOUI, A. EL BYED, **An Advanced Intelligent Support System for Multi-modal Transportation Network Based on Multi-Agent Architecture**, Advanced Intelligent Systems for Applied Computing Sciences, Volume 4 - Pages 98-106. 2020 https://doi.org/10.1007/978-3-030-36674-2_10
3. J. LARIOUI, A. EL BYED, **Towards a Semantic Layer Design for an Advanced Intelligent Multimodal Transportation System,** International Journal of Advanced Trends in Computer Science and Engineering, 9(2), March - April 2020, 2471 – 2478 https://doi.org/10.30534/ijatcse/2020/236922020
4. J. LARIOUI, A. EL BYED, **An Agent-based architecture for Multi-modal Transportation Using Prometheus Methodology Design**, International Journal of Natural Computing Research, Submitted
5. F. Lhafiane, A. Elbyed, M. Bouchoum, **Multi-Agent System Architecture Oriented Prometheus Methodology Design for Reverse Logistics**, World Academy of Science, Engineering and Technology International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol:9, No:8, 2015
6. L. PADGHAM, M. WINIKOFF: **Prometheus: a Pragmatic Methodology for Engineering Intelligent Agents**. J. DEBENHAM, B. HENDERSON SELLERS, N. JENNINGS et J. ODELL, éditeurs : Proceedings of the OOPSLA 02 - Workshop on Agent-Oriented Methodologies. COTAR, 2002.
7. L. PADGHAM et M. WINIKOFF 2003: **Prometheus: A Methodology for Developing Intelligent Agents**. Dans F. GIUNCHIGLIA, J. ODELL et G. WEISS, éditeurs : Agent-Oriented Software Engineering III, Third International Workshop, AOSE 2002, Bologna, Italy, July 15, 2002, Revised Papers and Invited Contributions, volume 2585 de Lecture Notes in Computer Science (LNCS), pages 174–185. Springer-Verlag, 2003.
8. L. PADGHAM et M. WINIKOFF, **Prometheus: A Practical Agent-Oriented Methodology**. In: Henderson-Sellers, B., Giorgini, P (Eds). Agent-Oriented Methodologies, pp. 107-135. IDEA Group Publishing , 2002 https://doi.org/10.4018/978-1-59140-581-8.ch005
9. R. Cunha, **Development of a Graphical Tool to integrate the Prometheus AEOlus methodology and Jason Platform**, ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal Regular Issue, Vol. 6 N. 4,41-54 , (2017).
10. J.-P Briot, Y. Demazeau, **Principes Et Architecture Des Systemes Multi-Agents**, Volume 217. Hermes Science Publications. 2001
11. G. Caire, W. Coulier, F. Garijo, J. Gomez, J. Pavón, F. Leal, P. Chainho, P. Kearney, J. Stark, R. Evans. **Agent Oriented Analysis Using Message/Uml.** In

Agent-Oriented Software Engineering Ii, Pages 119-135. Springer. 2002

12. J. Thangarajah, L. Padgham, M. Winikoff,, **Prometheus Design Tool**. In Proceedings Of The Fourth International Joint Conference On Autonomous Agents And Multiagent Systems, Pages 127-128. Acm. 2005 https://doi.org/10.1145/1082473.1082817
13. H. Sun, J. Thangarajah, L. Padgham, **Eclipse-Based Prometheus Design Tool,** Proc. Of 9th Int. Conf. On Autonomous Agents And Multiagent Systems, May, 10–14, , Toronto, Canada 2010
14. L. Padgham, J. Thangarajah, M. Winikoff, **The Prometheus Design Tool**, A Conference Management System Case Study, M. Luck And L. Padgham (Eds.): Aose 2007, Lncs 4951, Pp. 197–211, 2008. https://doi.org/10.1007/978-3-540-79488-2_15
15. D.C. Schmidt, **Model-Driven Engineering**. Guest Editor’s Introduction: Computer 39(2), 25–31 (2006)
16. M. Perepletchikov, L. Padgham: **Systematic Incremental Development Of Agent Systems Using Prometheus**. In: Fifth International Conference On Quality Software, Pp. 413–418 (2005)
17. A. Fernández-Caballero And José M. Gascueña, **Developing Multi-Agent Systems Through Integrating Prometheus**, Ingenias And Icaro-T, Icaart 2009, Ccis 67, Pp. 219–232, 2010 https://doi.org/10.1007/978-3-642-11819-7_17
18. W N Hussein et al., **A Methodology for Big Data Analytics and IoT-Oriented Transportation System for future implementation**, International Journal of Emerging Trends in Engineering Research, 7(11), November 2019, 449 – 453 https://doi.org/10.30534/ijeter/2019/087112019
19. Nishad Nawaz, **Artificial Intelligence Face Recognition for applicant tracking system**, International Journal of Emerging Trends in Engineering Research, 7(12), December 2019, 895 - 901