Chapter 1

Dynamic Car-pooling

1.1. Introduction

In order to mitigate the negative impact of private cars and thus heal the environmental image of a personal vehicle, car-sharing systems were born. Within this context, car-pooling in particular has been a notable success thanks to the contributions it brings mainly by reducing the number of cars on the road. Indeed, making the personal car a common mode of transport, car-pooling plays a role in the reduction of harmful gas emission rates. The contributions are quantifiable in terms of "non-emitted" CO_2, in addition to the many advantages it offers on both an individual and a collective level (e.g. reduction of budgets allocated for transport, time-space flexibility, comfort, social balance, etc.). Thus it has made its entrance into the field of research, and numerous systems have since emerged. Several studies have been conducted in a manner that draws on the fields of computer science, artificial intelligence, GIS (Geographical Information Systems), the Internet and telecommunications, etc. Making use of new technologies and the establishment of a more or less evolved system has precedence over any other goal in existing approaches. Web-based media are today operational and allow the

---

Chapter written by Slim HAMMADI and Nawel ZANGAR.

general public to register and benefit from fairly limited services such as publications and consultation of offers and demand, as well as acquisition of contact details for potential car-poolers. Unfortunately, this type of system is the only one that deserves recognition because the rest, despite their openness to advanced features such as the integration of real time and the automation of allocation tasks based on multi-agent systems, have remained at the "*idea*" or "*draft*" stage, and are not liable to improvement. Irrevocably launched onto the road towards improvement, in this chapter we propose the implementation of an optimized dynamic car-pooling system. Two main concepts will be discussed in particular, namely the modeling of the problem as a distributed dynamic graph, on the basis of which distributed software architecture is established, and the deployment of a multitude of autonomous entities under this architecture. The combination of multi-agent systems with the foundations for optimization has thus been put in the service of effectiveness of processing for the establishment of an approach in the context of distributed artificial intelligence.

1.2. State of the art

In recent years, the sharing of vehicles has become a remarkable phenomenon mainly due to the economic and environmental benefits it offers. Thus, people tend to go for the "self-shared" vehicles (self-service vehicles, car-pooling, etc.) and discard their own automobiles. In this context, there are nowadays several studies that revolve around the sharing of vehicles, allowing the completion of relatively efficient systems to exist in about 1,000 cities around the world already [WCC]. These systems provide more or less satisfactory services for users but are still in the early stages. The work done to date can be subdivided into two categories according to the criteria for booking management adopted within the system. The first category is based on a static booking management while the second deals in real time with a dynamic aspect. In what follows, we present a non-exhaustive list of what has been done in this regard by making a distinction between:

– operational systems implementing the concept of "Shared Vehicles" (we actually focus on car-sharing, car-pooling sites remaining, for the most part, open and non-optimized systems);

– the academic work done for modeling and optimization of such systems.

The existing operational systems are:

Static systems

– SEFAGE (SElbstFAhrerGEnossenshaft), which could be translated as "drivers club". This is the oldest car-sharing organization traceable in the literature. It was founded in Zurich, Switzerland in 1948. It was essentially a club where members came together to buy a car. Without any commercial purpose, the main objective was to offer the service of having a vehicle available when needed. As the initiators were not aware of its innovative characteristics, SEFAGE never developed further.

– Lilas [LIL]: car-sharing was strongly implemented in Lille in France through this service in which a booking is made in advance by phone or online using a member number. The member, who receives a monthly invoice, must book each time for at least one hour of service but has the freedom of choosing a car of their choice from a station of their choice. However, despite the variety of stations and vehicles it provides to customers, the Lilas system has the disadvantage of limiting the time of use of its vehicles, working in a loop (return by the user to the station of departure) and not having real-time reservation services. Lilac offers competitive pricing for users subscribed to local transport and for families, and combines well with other forms of transport.

– Modulauto is the name associated with the car-sharing service available to the inhabitants of the city of Montpellier and the town of Nîmes in France. As part of the *France Auto-partage* service, Modulauto offers its members a fleet of self-service vehicles. Users reserve a car online or by phone at least 30 minutes in advance and the vehicle must be returned to a Modulauto station.

– Mobility is the market leader for car-sharing in Switzerland, with a portfolio of 55,500 clients for which it provides 1,700 vehicles. A partnership with public transport has contributed largely to the development of Mobility.

– Communauto [BEN] in Canada (Quebec, Montreal, Sherbrooke, Gatineau): since its foundation in 1994, Communauto appears to be a pioneering enterprise in America, as the manager of the oldest and one of the most important car-sharing services that has emerged on its side of the Atlantic. Communauto, which now has more than 14,000 members in Quebec, is the first car-sharing organization in the world to have signed the Charter for Sustainable Development of the International Union of Public Transport (IUPT).

– City Car Club presents itself as the solution to problems faced by owners of private vehicles in the UK. It is a car-sharing service offered to British citizens and, like its predecessors, has a fleet of self-service vehicles that members share amongst themselves.

– Cambio Stadt [MOB] in the city of Bremen in Germany: implemented in 2002 and owing to a partnership with Vivaldi and cooperation with Civitas, during its first three years, this project experienced a 43% increase in the number of users (2,455 to 3,512 in January 2005).

– I-GO and zipcar USA: having the same principle as the majority of the various systems listed above, car-sharing in the United States presents itself, for the zipcar company founded in June 2000 [ZIP] and I-GO [IGO], as a system of advance booking of self-service vehicles of limited use duration. These vehicles are collected from stations designed for this purpose and delivered back to the starting station after use.

The static systems described above require advance booking based on static reasoning, without considering instantaneous events. Thus, these systems do not provide real-time vehicle allocation, nor do they provide an immediate response to the user. To remedy this deficit, dynamic car-sharing systems have emerged with real-time service management.

Dynamic systems

– PRAXITELE: the PRAXITELE Research and Development program was founded in 1993 based on a consortium from the industry (CGEA, Renault, the Dassault Electronics Group and EDF)

and two research institutes (INRIA and INRETS). The service became operational in October 1997 in Saint-Quentin-en-Yvelines with 50 cars (PRAXICARS) and five carparks (PRAXIPARKS). PRAXITELE is a new form of public transport complementary to public transport and taxis. This new service is primarily intended for journeys and schedules when demand is diffuse and public transport is inadequate in terms of frequency and profitability. Its use is restricted to a geographical area ranging from a few hundred meters to several kilometers in some cases. To use PRAXITELE, one must first become a member of the service. A "credit card"-type memory card using contactless technology, the PRAXICARD, is then delivered to the client allowing them to access the cars, consult the information terminals installed in the stations and carry out payment. The PRAXIPARK stations act both as an interface between the system and its users through information terminals (information exchange with PRAXICENTRE, controls), and as a local operation system through its chargers (for recharging vehicles, etc.). The PRAXICAR vehicles use electric engines and are equipped with automatic chargers which charge by induction, facilitating their use, management and operation. They include an on-board computer which controls the opening in order to collect payment and ensure a dialogue with the user. Currently, the vehicles are electric versions of existing models (Renault Clio). Eventually, the idea is to use the concept of small electric cars, with dimensions better suited to urban use.

– LISELEC [LIS]: created by PSA, VIA GTI and Alcatel CGA Transport, LISELEC is part of the global travel policy that has been leading the the La Rochelle agglomeration for many years. The experiment began in September 1999 in partnership with: the General Council of Charente-Maritime, the Regional Council of Poitou-Charentes, ADEME Poitou-Charentes, EDF and PREDIT. The service offered by LISELEC works similarly to the PRAXICENTRE system mentioned above, also with 50 electric cars (106, Saxo, Berlingo and Gem) dispersed across seven stations. These vehicles are available at any time of the day or night with a membership contract. With LISELEC, parking is free in La Rochelle and in the spaces reserved in stations. This system is intended for all city residents and for all regular visitors for short urban journeys and additionally offers preferential tariffs for long-distance travel compared to a vehicle

rented from a classic car hire company (National/Citer). At the end of April 2002, there were 485 members with an average increase of six new members per week.

– Cité Vu from VU LOG: VU LOG is a company offering a different take on car-sharing in downtown Antibes. It presents a new car-sharing system (Summer 2008) called "Cité Vu" which offers its members a fleet of public individual electric vehicles. These are organized into self-service with instant access and do not require prior reservation. Members of this service therefore benefit from an ease of access as well as the possibility of parking anywhere in the city without being obliged to return the car to a preset station. However, the vehicles available are only designed for short-distance urban travel, traveling at a maximum speed of 45 km/h.

– AutoLib [WCC]: in the same context as "Cité VU" by VU LOG, a new project named AutoLib was launched in December 2011. This project takes into consideration the management of reservations and customer requests in real time. AutoLib should make available to prospective members a fleet of electric vehicles, thereby reducing CO_2 and greenhouse gas emissions.

– Telebus (On-Demand Transport for the disabled [BOR 97]): this TAD system, established in Berlin, Germany in 1981, has the interesting characteristic of having been the testing ground for methods of "smart" control whose application in the years 1990/2000 has increased covered demand by 40%.

Academic points of view

The *universities of Quebec* can claim a certain leadership regarding the design of computer systems to support the decision for mobility and transport: CRT MONTREAL, GERAD, CIRRELT Laboratory, GIRO Society. They have extensively studied conventional routing problems, load planning and allocation in static and dynamic contexts [BAR 07; COR 02]: *Pick up and Delivery*, *Dial and Ride* [COR 03; COR 07; CED 01]: *Crew scheduling*, using linear programming techniques [BAR 98] (column generation, etc.) now commonly used in Air France, SNCF, EDF and France Telecom R/D.

However, these problems, since the history of Decision Support, have also been studied in Europe, particularly in Italy (La Sapienza University and I.P. Rome for the *Dynamic TAD*, I.P. MILAN for *Intelligent Car-sharing*), Belgium (ULB for *Network Design* and *Pricing*), Germany (Bonn, Berlin, Osnabruck, Saarbrücken, Kaiserslautern, etc. for the coupling between Mobility and Industrial Planning), and France (UTT, UTC HEUDYASIC: *Tours Planning*, *Network Design*; IRCYN NANTES: *Hospital Mobility*; LAGIS Ecole Centrale of Lille: *Multimodality*, and at the industrial level, in DER AIR France or SSII EURODECISION, ILOG/IBM, AERTLIS, etc.). If one of the current trends is to encourage innovative mobility and new vehicle categories (INRIA, LASMEA, CRYSTAL ANR Project), another is to consider it through a systematic perspective by linking it to the concept of sustainable economy and environmental concerns [COR 03]. Also, a request could be made to schedule days such that production operations and transport activities related to internal enterprise logistics are coupled [MCB 83; SCHU 78; POC 06].

The adaptation of algorithms from a static context (point of view of most programmers) to a dynamic context has often been studied, but most come with the cost of not considering how users, vehicles and controllers actually collaborate through the communication architecture [JES 08; LEC 06; PAS 95]. Studies on the statistical analysis of the demand for mobility were carried out on a macro level during the 1980s (Logit TERESE of CERTU model and INRETS, [TER]).

These models do not address the question of the susceptibility of demand towards a micro-system (TAD, etc.) in case of changes in its configuration [JÄG 03]. Interesting studies have been conducted in this regard at the University of Montreal, around the management of a fleet of ambulances for emergencies [BRO 03], which involves a spatio-temporal model, and at MIT on the elasticity of demand for public transport [ASH 98]. Regarding *Pricing*, the most innovative experiments have involved *Cooperative* or *Non-Cooperative Game Theory* models, intended to enable the identification of stable market shares in multimodal transport [ALT 04; BOU 02; TAM 91], as well

as *bi-level optimization* models for combining pricing with a control procedure (decongestion) for access to a network.

The Optimization of Networks has been promoted widely by the deregulation of energy production markets, transport and telecommunications, initially with a focus on issues related to traffic stability and the consideration of security constraints [HU 05; WAR 52]. The emergence of embedded technologies allowing the planning of real-time networks (*SmartGrids*, etc.) is leading to a reformulation of classical problems for very large amounts of data on dynamic networks (with time indexing nodes) and stochastic demands. The case of data acquisition and monitoring systems (Sensor Networks) is an especially rich case [FLE 09].

Very recently, prospective studies were conducted in California and The Netherlands on the potential opportunities offered by the introduction of Web technology and distributed in the management of mobility support systems [VAN 04; GIL 02; HEU 08; HIG 00; STO 00], with a focus on the design of advanced system/user communication systems, the management of human resources and the control of safety. Studies have also been conducted on the prospect of the increased flow of automated vehicles in mobility and transport services [KON 06].

1.3. Complexity of the optimized dynamic car-pooling problem: comparison and similarities with other existing systems

In order to properly identify the complexity of the car-pooling problem in terms of optimization difficulty, two types of optimization problem are considered: the first concerns the combinatorial optimization problem for which there is no accurate and fast algorithm. We can cite a known problem, which is that of the traveling salesman, classified as “NP-Hard” whose exact solution can only be done by a time calculation proportional to N^n, where n is an integer and N is the number of unknowns. The second type relates to optimization problems of which the variables are continuous. For this type of problem, there are no algorithms guaranteeing certainty of finding a global optimum with a finite number of calculations.

A comparative study on the problem of car-pooling with dynamic optimization, CDO, with other optimization problems known for their complexities has been conducted. Indeed there are many similarities between the CDO dynamic car-pooling problem and the traveling salesman problem, TSP, such as the nodes (origin/destination vs. cities), and the objective function (distance vs. time, distance, CO_2 emission). The CDO is hence considered an NP-Hard combinatorial problem whose complexity is expressed in a manner similar to that of the TSP.

$$(O(CDO) \equiv O(TPSP))$$

In addition to the traveling salesman problem, we can also refer to the vehicle routing problem (VRP) and in particular the collection and delivery of objects (i.e. products, goods, people, etc.). A comparative study of the CDO problem with two variants of DPDPs (Dynamic Pickup and Delivery Problems), which are the Swapping Problem [ANI 92; COR 07], SP, was performed. This study showed that there was even more similarity than with the TSP, bearing in mind that the VRP problems are known for their high complexity of exponential order. By analogy to these problems, the CDO is thus stated as an NP-Hard combinatorial optimization problem

$$O(CDO) \equiv O(Swapping\ Problem) \equiv (Dynamic\ DARP)\ .$$

hence the great difficulty in solving the CDO problem as well as establishing an efficient optimization system.

1.3.1. *Graphical modeling for the implementation of a distributed physical architecture*

For the graphical modeling of the car-pooling system, we investigated works in the context of the vehicle routing problem [IOR 07] and in particular the problems of collection and real-time distribution [BER 10; WU 09]. Indeed, the random and dynamic aspects of events and data [SAV 95; XIA 08] defining these problems join the randomness of the supply and demand of the real-time car-pooling problem. The transport problem of on-demand transport for disabled persons [ATA 10; COR 03], which consists of adjusting the supply according to the demand of users, is considered here,

especially since it is even closer to our problem, given that we place ourselves in the context of transportation of people and not of goods.

Bearing in mind the very random aspect of supply and demand by users, the architecture of the proposed system must be able to monitor developments in real-time and will be based on the concept of a dynamic distribution graph. Thus, for n users issuing car-share requests and m vehicles offering trips, whether they are already in circulation or not at a given time, t, we define the dynamic car-pooling system as the triplet $T(t) = (G(t), R(t), O(t))$. In this expression, each parameter is relative to a particular component that contributes to the successful implementation of the system:

– $G(t) = (N(t), A(t))$ is a dynamic directed graph representing the served network where all offers and requests of users of the service are scattered.

– $N(t)$ is the set of nodes constructed based on geographical positions of users of the service, whether as car-sharers or car-sharees, and the details of the required or expected travel (i.e. benchmarks and routes to be taken).

For reasons of safety and security, the geolocation of users (i.e. pedestrians and cars) is done in real time thanks to a positioning module integrated within the system.

$A(t)$ is the set of arcs on the graph, which are formed as a result of the extraction of required information on what is available at a given time, t. They essentially represent the routes that will be served by the cars and therefore correspond to the journey remaining to be traveled by drivers during the trip or not. These routes are constructed and updated in real-time based on the time when a vehicle passes a check-point chosen by the system. This representation must be consistent with the nature of the roads as well as the possibilities for using them (e.g. one-way street, two-way street, etc.). An arc can be oriented either in one direction or in both if two or more cars have used the same section of road in opposite directions.

Hence the itineraries of cars are composed of several sections. Two or more cars can share more than one road in the same direction. Thus defined, these sections are the arcs of the graph $G(t)$ connecting the nodes between them, thus identifying their orientation according to the directions in which represented roads are used. The end nodes of these segments of journeys can be the origin or destination or both at once.

Each node can have one or more successors and hence represents an intermediate or final destination of the vehicle(s) or user(s).

We define the set of successor nodes of x written $N_x^+(t)$ as follows:

$$N_x^+(t) = \{v \in N(t) / x \neq v \text{ and } (x,v) \in A(t)\} \quad [1.1]$$

Similarly, each node can have one or more predecessors which are the original ends (i.e. origins) of arcs whose destination node is considered.

The set of predecessor nodes of x, written $N_x^-(t)$, is defined as follows:

$$N_x^-(t) = \{v \in N(t) / x \neq v \text{ and } (v,x) \in A(t)\} \quad [1.2]$$

1.3.2. *Collection of requests for car-pooling and data modeling*

Given that the random nature of our graph relies heavily on data that is constantly changing, we propose in the following to see how car-pooling requests will be structured, and data modeling.

We consider $R(t) = \bigcup_{p=1}^{n} \{R_p(t)\}$ the set of n requests issued by n users of the service $U(t) = \bigcup_{p=1}^{n} \{U_p(t)\}$.

Only users logged into the system at time *t* and who have issued car-pooling requests are considered.

1.3.2.1. *Structure of a request for car-pooling*

The structuring of car-pooling requests is done as follows:

$$R_p(t) = \left\{ R_p^+, R_p^-, P_p, \left[d_p, a_p \right] \right\} \quad [1.3]$$

Each request is specific to a particular user, U_p, indicating their travel preferences:

– R_p^+ indicates the geographical coordinates of the desired origin of travel. If the user fails to specify their point of departure, geographical positioning at time t is captured and will be considered the original source of the route to follow.

– R_p^- is the final destination that the user wishes to achieve. Introduced in a similar manner as at the beginning of the planned route, the destination can be inserted as an address or geographic coordinates.

The origins and destinations of trips requested by users are represented in the proposed model by pairs of nodes on the graph, $G(t)$. Consequently, $N_R(t)$ is the set of nodes in $N(t)$ relative to requests from the car-sharees (i.e. pedestrians, passengers):

$$N_R(t) = \bigcup_{p=1}^{n} \left\{ R_p^+, R_p^- \right\} \subseteq N(t) \quad [1.4]$$

– P_p is the number of people, U_p included, to transport from the origin R_p^+ to the destination $R_p^- \left(P_p \geq 1 \right)$. By default, this parameter equals 1, considering U_p as a single passenger.

– $[d_p, a_p]$ designates a limited time interval that defines the overall length of the maximum route tolerated by the user [DUM 91], where d_p is the earliest possible start time for the user, p, and a_p is the latest tolerated arrival time. Otherwise, failing to have specified

preferences, the user will be assigned default values. These values correspond to the system time relative to the time of transmission of the request for the earliest departure time and the Estimated Time of Arrival (ETA) for the latest arrival time:

$$ETA(R_p^+, R_p^-, t) = \rho_{R_p^+, R_p^-}(t) + \tau(R_p^+, R_p^-) \quad [1.5]$$

This time is calculated based on the two following functions:

1. A weighting function calculates the journey time $(\rho_{O,D}(t))$ required to travel a set route from an origin O to a destination D. In this case, the graph $G(t)$ is said to be "valued", allowing the assignment of a value (i.e. a weight) to each arc of the graph. The weight of an arc (O,D) such that $(O,D) \in A(t)$ equals $(\rho_{O,D}(t))$ is calculated as a function of several factors:

$$\rho O,D(t) = \text{Distance}\,(O,D) * \frac{1}{AJS(O,D,t)} \; with \; (O,D) \in A(t) \quad [1.6]$$

One must note that in the context of urban and pre-urban transport, the weightings of the arcs cannot be negative [FEK 10]. Thus, $\forall (O,D) \in A(t), \rho_{O,D}(t) \geq 0$.

The weighting of arcs of the graph is calculated based on:

– the function that calculates the distance between two geographical points represented by their GPS coordinates (i.e. longitude, latitude). In this formula, the spherical law of cosine is taken into account and is as follows:

$$\text{Distance}\,(O,D) = \text{acos}\begin{pmatrix} \sin(\text{latitude}_{\,O}) * \sin(\text{latitude}_{\,D}) + \cos(\text{latitude}_{\,O}) * \\ \cos(\text{latitude}_{\,D}) * \cos(\text{longitude}_{\,D} - \text{longitude}_{\,O}) \end{pmatrix} * R \quad [1.7]$$

– the AJS parameter, which indicates the Average Journey Speed on a specific route at a specific time t. This parameter varies depending on several factors, amongst others the nature of the road, weather conditions, rush hour, etc.

2. The $\tau(O,D)$ function calculates the maximum possible delay on a route $[O,D]$. It is not considered a route segment unless there is an arc $(O,D) \in A(t)$ connecting both ends in the direction indicated. The rate of tolerable delay is calculated as a function of the distance between the origin and destination nodes of the route proportional to a Tolerable Delay Threshold (*TDT*), configurable by the system administrator:

$$\tau(O,D) = \text{Distance } (O,D) * \frac{\alpha}{\beta} \qquad [1.8]$$

TDT represents the value of tolerable delay (α) in minutes for a ride of (β) kilometers, to calculate the latest arrival time at the end of the route in question.

1.3.3. *Matrix structure to collect information on requests*

In what follows, a matrix structure is proposed for the collection and representation of information:

$$D(t) = \begin{array}{c} \\ 1 \\ \vdots \\ i \\ \vdots \\ j \\ \vdots \\ k \end{array} \begin{array}{c} \begin{array}{ccccccc} 1 & \cdots & i & \cdots & j & \cdots & k \end{array} \\ \begin{pmatrix} \phi & & \cdots & & \cdots & & \cdots \\ & \ddots & & & & & \\ \cdots & & \phi & & D_{i,j} & & \cdots \\ & \cdots & & \ddots & & \cdots & \\ \vdots & & D_{j,i} & & \phi & & \vdots \\ \cdots & & \cdots & & \cdots & \ddots & \cdots \\ \cdots & & \cdots & & \cdots & & \phi \end{pmatrix} \end{array}$$

Each element $D_{i,j}$ with $1 \le i, j \le k$ of the matrix $D(t)$ is a model of the existing demand on a set route $[a,b]$ starting from origin a going to destination b. The indices of the matrix refer to the nodes of the network. A list of variable size is used within each element of the matrix to represent the set of requests relative to the route represented

by the indices of the element. In these elements, we find users who have requested to travel this route meaning that users whose entire requested trip $[R_\rho^+, R_\rho^-]$ or portion of the trip concerns the route $[a,b]$. These users are represented with all the specifics of travel required on the section in question (i.e. the preferred time interval and the number of passengers).

$$D_{i,j} = \bigcup_{p=1}^{n} \left\{ \left(U_p, P_p, \left[d_p, a_p \right] \right) \right\} \text{ such that } [a,b] \in \left[R_p^+, R_p^- \right] \quad [1.9]$$

We denote $O(t) = \bigcup_{j=1}^{m} \left\{ O_j(t) \right\}$ as the set of offers of m vehicles proposed for travel at time t.

$V(t) = \bigcup_{j=1}^{m} \left\{ V_j(t) \right\}$ is the fleet of vehicles having offered a bid for a specific trip.

To any specified car-sharing trips offered corresponds a couple of nodes in $N(t)$ representing its starting point (i.e. origin (O_j^+)) and destination (i.e. destination (O_j^-)):

$$N_o(t) = \bigcup_{j=1}^{m} \left\{ O_j^+, O_j^- \right\} \subseteq N(t) \quad [1.10]$$

$N_o(t)$ denotes the set of origin and destination nodes indicated by drivers of vehicles submitting car-pooling offers at time t.

Structure of a car-pooling offer

Offers of vehicles are modeled by our system in a similar manner to car-pooling requests previously described, as in the equation below:

$$O_j(t) = \left\{ O_j^+, O_j^-, l_j \begin{pmatrix} ID_1 \\ \cdots \\ ID_z \end{pmatrix}, \left[d_j, a_j \right] \right\} \quad [1.11]$$

$O_j(t)$: offer of journey relative to car V_j

To submit an offer, all drivers must specify:

– $\left(O_j^+\right)$ as the starting point of their journey;

– $\left(O_j^-\right)$ as the destination;

– l_j represents the number of spaces available in the vehicle;

– $[d_j.a_j]$ defines the travel preferences of the car driver;

– the vector $ID_j = \begin{pmatrix} ID_1 \\ \vdots \\ ID_z \end{pmatrix} = \left(ID_{l,j}, \ldots, ID_{q,j}, \ldots, ID_{z,j} \right)^T$ indicates the set of Intermediate Destinations (*ID*) through which vehicle V_j must pass, from which time taken, *t*, to reach its final destination O_j^- is considered;

– we define $N_{ID}(t)$ as being the union of each one's intermediate destinations:

$$N_{ID}(t) = \bigcup_{j=1}^{m} \bigcup_{q=1}^{z} ID_{q,j} \subseteq N(t) \quad [1.12]$$

We then define $N(t)$ as follows:

$$N(t) = N_R(t) \bigcup N_O(t) \bigcup N_{ID}(t) \quad [1.13]$$

1.3.4. *Matrix representation for modeling car-pooling offers*

In order to take into account all existing car-pooling offers, we propose to model the offers of all vehicles of the fleet in an Origin Destination matrix $I(t) = (r_{o,d})_{1 \le o,d \le f}$.

The indices o and d of elements $I(t)$ here refer to the nodes of origin and destination of a route $[a,b]$.

Each index represents a node that can be either origin or destination or both at the same time, representing in turn different segments of the route (e.g. $[a,b]$ or $[b,a]$) depending on the offer being considered.

This matrix representation is based on a decomposition of the set of overall itineraries $[a,b]$ on elementary roads such that $(a,b) \in A(t)$. This decomposition is a figure of disassembly of vehicle routes based on their origins, destinations and visited points:

$$\begin{aligned} &a \in \{N_O(t) \cup N_{ID}(t)\} \text{ and } b \in \{N_O(t) \cup N_{ID}(t)\} \\ &\forall o \in \{1...f\} \text{ and } \forall d \in \{1...f\} \end{aligned} \quad [1.14]$$

The indices o and d of matrix $I(t)$ refer to the set of nodes where $f = \left|N_o(t) \bigcup N_{ID}(t)\right| = Card(N_o(t) \bigcup N_{ID}(t))$, wherefrom $f \le 2m + Z$, with $Z = \left|N_{ID}(t)\right| = Card(N_{ID}(t))$. The set of offers are thus reduced to a matrix representation of the sets of routes taken, considering down to the smallest part of route $[a,b]$ defined by an origin a to a destination b:

$$I(t) = \begin{array}{c} \\ 1 \\ \vdots \\ o \\ \vdots \\ d \\ \vdots \\ f \end{array} \begin{array}{c} \begin{array}{ccccccc} 1 & \cdots & o & \cdots & d & \cdots & f \end{array} \\ \begin{pmatrix} \phi & & \cdots & & \cdots & & \\ & \ddots & & & & \cdots & \\ \cdots & & \phi & & r_{o,d} & & \phi \\ & \cdots & & \ddots & & \cdots & \\ X & & r_{d,o} & & \phi & & \\ \cdots & & & & & \ddots & \\ & \cdots & & X & & & \phi \end{pmatrix} \end{array}$$

Each element $r_{o,d}$ of the matrix represents the supply side of the route section represented by $[a,b]$. The indices of the matrix, being represented by the overall set of origins, destinations and intersection points of all cars, allow route sections to be represented in the same matrix without being used or even allowing passage in the direction indicated. The matrix elements can, as the case may be, have one of the following values:

$$r_{o,d} = \begin{cases} X : \text{if according to the highway code, } [a,b] \text{ cannot be inserted in the direction indicated,} \\ *: \textit{if } [a,b] \text{ is practically doable but no offer exists (i.e.no vehicle covers this area)} \\ \bigcup_{j=1}^{m} \left\{ \left(Vj, l_{a,b}^{j}, \left[d_a^j, a_b^j \right] \right) \right\} \text{such that} [a,b] \in \left[O_j^+, O_j^- \right] : \text{The set of vehicles intersecting route} \\ [a,b] \text{ with, for each ,different displacement parameters} \end{cases}$$

$r_{o,d}$ contains the list of parameters specific to each vehicle serving the section of journey in question, namely:

– $l_{a,b}^{j}$ is the number of spaces available in vehicle V_j on route $[a,b]$;

– $\left[d_a^j, a_b^j\right]$ indicates for all vehicles V_j serving $[a,b]$, the earliest starting time d_a^j from the starting point *a* of $[a,b]$ and the latest arrival time a_b^j at the end point *b*.

The elements of the matrix $I(t)$ are relative to route sections being used or not by one or more vehicles. These routes are thus obtained by a decomposition of offered trips.

Any vehicle V_j of the fleet is associated with an overall itinerary IT_{V_j}, constructed in real time at the considered time *t*. Subsequently, $\forall j \in \{1..m\}$, is defined as a string of ordered nodes ($v_0, v_1, ...v_n$), with $n = z + 2$ and $z = \left|ID_j\right| = Card\left(ID_j\right)$:

$$IT_{V_j}(t) = (v_0, v_1, ..., v_n) \text{with} \begin{cases} v_k \in N(t), \forall k \in [0, n] \\ v_0 = O_j^+, v_n = O_j^-, \bigcup_{k=1}^{n-1} v_k = ID_j(t) \\ (v_k, v_{k+1}) \in A(t), \forall v_k, v_k + 1 \in IT_{V_j}(t); k \in \{0..n-1\} \end{cases}$$

Every two successive nodes are connected together by an arc in $A(t)$ and thus the overall itinerary of the car is defined as the adjacent routes starting from the origin of the journey and going towards the defined end point. In the following formula (equation [1.15]), an itinerary IT_{V_j} is defined as a junction of n – 1 Partial Itinerary (*PI*) so long as it consists of n successive nodes:

$$IT_{V_j}(t) = \left(PI_1^j \oplus ... \oplus PI_{n-1}^j\right) = \left(\left[O_j^+, ID_1\right], ..., \left[ID_{q-1}, ID_q\right], ..., \left[ID_z, O_j^-\right]\right)$$

[1.15]

PI_s^j represents the Partial Itinerary number s in the overall itinerary $IT_{V_j}(t)$ of V_j; $\oplus$ is used here as a juxtaposition operator to show the succession of route sections designated by *PI* in a fixed order such as to form the overall itinerary. Each *PI* is modeled by its corresponding element in the matrix $o \in N_h^+(t)$; it has its own properties and can include one or more offers.

In addition, for each offer on a particular *PI*, the parameters specific to it are automatically calculated by the system. Indeed, the number of spaces available is updated at each pick up and drop off of a passenger. It is also possible to calculate the time interval extremities for the start and end on any part of route $[o,d]$, as soon as the necessary information on the time course of the preceding route segment is obtained. By "necessary information", we specifically mean the earliest departure time and the latest arrival time of vehicle V_j on the route section $[h,o]$,which directly precedes $[o,d]$.$[h,o]$ precedes $[o,d]$ if, $o \in N_h^+(t)$, $d \in N_o^+(t)$, $(h,o) \in A(t)$, $(o,d) \in A(t)$ and: $\left[[h,o],[o,d]\right] \subseteq IT_{V_j}(t)$

$$\begin{cases} d_o^j(t) = d_h^j(t) + \rho_{h,o}(t) = \text{the earliest arrival time at o} \\ a_d^j(t) = d_o^j(t) + \rho_{o,d}(t) + \text{Distance}(O_j^+, d) * \dfrac{\left(a_{o_j^-} - d_{o_j^-}\right)}{\text{Distance}\left(O_j^+, O_j^-\right)} \end{cases}$$

$\tau(o,d) = \text{Distance}\left(O_j^+, d\right) * \dfrac{\left(a_{o_j^-} - d_{o_j^-}\right)}{\text{Distance}\left(O_j^+, O_j^-\right)}$ is the possible delay in arriving at d from o. This delay is estimated based on the delay tolerated by user $\left(a_{o_j^-} - d_{o_j^-}\right)$ across the entire trip. We then give $\alpha = a_{o_j^-} - d_{o_j^-}$ the value of the delay that the driver can tolerate for

for their overall journey and $\beta = \text{Distance}(O_j^+, O_j^-)$. $a_{O_j^-}$ and $d_{O_j^-}$ respectively express the latest arrival time set by the user and the earliest time of departure (arrival) to O_j^- .

1.3.5. *Modeling constraints of vehicles' allocation to users*

In what follows, we consider an approach that suggests considering the individual travel preferences of the car-pool applicant as well as the car-pool drivers offering the trips, whose parameters are set in advance but still somewhat flexible. It is therefore important to consider a certain degree of flexibility in relation to the construction of itineraries in order to meet travel requirements.

This results in the creation of an itinerary composed of several sub-itineraries, to which different vehicles are assigned each time. Figure 1.1 shows an example of a route composition solution for the user U_i from the available fleet where several vehicles can be considered if a single vehicle cannot alone meet the overall request.

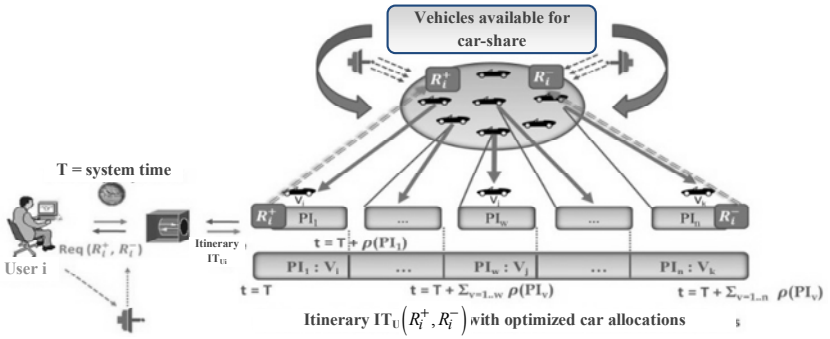

Figure 1.1. *Composition of solutions based on allocation of available supply*

Multiple offers on the same route allow us to choose the one that best minimizes the overall length of the trip: an optimization criterion chosen at system level. Thus, the car that best optimizes the overall travel time on each section in question is ultimately selected to build the final solution:

$$S_{U_i}(t) = (S_{1,i} \; o \cdots o \; S_{w,i} \; o \cdots o \; S_{n,i}) \qquad [1.16]$$

where $S_{U_i}(t)$ is a comprehensive optimized solution provided to the user, U_i, at the end of the processing of request $S_{w,i}$. This solution is, as equation [1.16] shows, reconstructed from a sequence of several partial solutions o_j that are relative to each part of the itinerary to be taken by U_i in order to achieve their goal. Here, "o" is a combination operator of partial solutions $R_i(t)$, indicating compliance with the order and direction in which the solutions are displayed, and with the image of the sequence of partial itineraries $\left(PI_{w,i}\right)$ relative to them. Each partial solution $S_{w,i}$, $\forall w \in \{l..n\}$, is then modeled as a data structure containing the necessary information for the route section in question. Aside from the pick-up $\left(PI_{w,i}\right)$ and drop-off $\left(PI^{-}_{w,i}\right)$, this structure provides data essential to the solution, namely, details about the vehicle in question (V_j), the earliest departure time and latest end time on the extremities of the itinerary:

$$S_{w,i} = \left(PI^{+}_{w,i}, PI^{-}_{w,i}, V_j, d_{PI^{+}_{w,i}}, a_{Pi^{-}_{w,i}} \right) \qquad [1.17]$$

The solution itinerary provided to the user U_i is then formulated as follows:

$$IT_{U_i}(t) = \left(PI^{v}_{l,i} \oplus \cdots \oplus PI^{j}_{w,i} \oplus \cdots \oplus PI^{k}_{n,i} \right) \qquad [1.18]$$

Where $PI^{j}_{w,i}$ is the w^{th} Partial Itinerary served by vehicle V_j in the overall route defined as a result of processing the request of user *i* (i.e. $R_i(t)$). Here, *j* refers to the car that has the optimum journey time out of V_w cars on the route section in question, $PI^{j}_{w,i}$. V_w is the set of vehicles whose respective itineraries include the path $PI^{j}_{w,i}$ in the

direction indicated, whilst satisfying the correspondence constraints and thus forming a set of "*feasible*" solutions for the considered route section.

Thus, partial itineraries for common routes are processed firstly in such a way as to extract the set of feasible solutions, meaning the cars that intersect the corresponding routes in the appropriate direction, in the desired time period and providing a sufficient number of spaces. Next, the car with optimal travel time is selected.

The algorithms we propose are executed following two main steps: firstly, we start by finding the set of feasible solutions V_w and secondly, the optimal solution among them.

To be considered feasible, a proposal for an offer O_j on a stretch of route $PI_{w,i}$ must satisfy a set of constraints that match the following criteria:

– $d_{PI^j_{w,i}} \geq a_{PI^v_{w-1,i}} - \tau\left(PI^+_{w,i}, PI^-_{w,i}\right), \forall w \in \{2..n\}$: a vehicle V_j can only be considered in the allocation process if it satisfies the first corresponding constraint (i.e. matching $PI_{w,i} \in IT_{V_j}$) between its overall itinerary and $PI_{w,i}$. This is reflected by the fact that V_j must travel its departure route section $l^j_w \geq P_i$ to its final destination $\left(PI^-_{w,i}\right)$ in the appropriate direction.

– $l^j_w \geq P_i$: the vehicle V_j is only a feasible solution if it provides a number of vacancies greater than or equal to the number of passengers seeking to travel on the route $PI_{w,i}$ as indicated by the request inserted by user U_i.

– $d_{PI^j_{1,i}} \geq d_{U_i}$: the earliest start time specified by the driver of V_j on the first portion of route $PI_{1,i}$ in IT_{U_i} must match that specified by U_i. The route section $PI_{1,i}$ goes from the origin of travel of passenger U_i, thus forming the initial part of the overall itinerary.

– $d_{PI^{j}_{w,i}} \geq a_{PI^{v}_{w-1,i}} - \tau\left(PI^{+}_{w,i}, PI^{-}_{w,i}\right), \forall w \in \{2..n\}$: moreover, for the rest of the sections that can be combined to make the overall itinerary, a car is only considered feasible on a part of route $\left(PI_{w,i}\right)$ if its earliest departure time from $\left(PI^{+}_{w,i}\right)$ is greater than or equal to the earliest arrival time at the end of the route that directly precedes it $\left(PI_{w-1,i}\right)$, regardless of which vehicle *v* is assigned to it (bearing in mind that $PI^{+}_{w,i} = PI^{-}_{w-1,i}$).

– $a_{PI^{j}_{n,i}} \leq a_{U_i}$: this final constraint particularly concerns the latest arrival time by vehicle V_j that can be assigned to the last stretch $\left(PI_{n,i}\right)$ of user *i*'s itinerary. The proposed system must meet the margin of delay tolerated by the user U_i. A vehicle V_j can only be assigned if it reaches the end of the course at a time less than or equal to the latest arrival time specified by U_i. Consequently, the vehicle V_j must reach the final end of the last section of route $\left(PI^{-}_{n-1,i} = d^{-}_{U_i}\right)$ of itinerary $IT_{U_i}(t)$ at a_{U_i} at the latest.

The verification of these constraints is done for all car-pooling offers issued at a time *t* in order to extract those that are likely to be considered as potential solutions to the request being processed. The set of feasible solutions on a part of the constructed itinerary is progressively obtained. This set, called V_w, refers to all vehicles serving the route labeled *w* of the itinerary of user U_i, while satisfying all the constraints listed above.

Since we mainly focus on our approach of optimization solutions provided to the user, a single solution is chosen for each route *w* in such a way as to optimize the itinerary. The criteria we have set refer to the Global Trip Duration (*TD*) which is calculable in terms of waiting time and travel time. These are, in turn, calculated using the weighting function (ρ) defined above. A fitness function allows

aggregation of waiting times ($W_{j,w}$) and travel ($T_{j,w}$) on a route section ($PI_{w,i}$), thus allowing the calculation of travel time for any vehicle $V_j \in V_w$:

$$Fitness = TD^{j}_{PI_{w,i}} = \omega_{1_i} W_{j,w} + \omega_{2_i} T_{j,w} \text{ with } V_j \in V_w \quad [1.19]$$

ω_{1_i} and ω_{2_i} are the aggregation parameters with a weighting of $W_{j,w}$ and $T_{j,w}$ respectively. They are set by the user and indicate their travel preferences by favoring optimization relative to waiting rather than travel time or spending less time in a car, even if it affects the waiting time.

The final solution $TD^{*}_{PI_{w,i}}$ is that which minimizes the fitness function on the section w, based on the objective function defined as the following:

$$TD^{*}_{PI_{w,i}} = Min_{V_j \in V_w} (TD^{j}_{PI_{w,i}} = \omega_{1_i} W_{j,w} + \omega_{2_i} T_{j,w}) \quad [1.20]$$

Based on the overall objective function thus defined and jointly with the principle of recomposing a complete solution from several partial solutions, the solution that optimizes the overall itinerary $\left(IT_{U_i}\right)$ is the one that optimizes the combination of all sections $w \in \{1..n\}$ of which the intersection constitutes the route $\left[R_i^+, R_i^-\right]$.

Given that our initial objective is to optimize the length of the entire journey carried out by U_i, the final solution is therefore to consider within each section $w \in IT_{U_i}$ the vehicle minimizing travel time. The sum of the optimized lengths on all sections of the itinerary constituting the full itinerary consequently leads to the minimization of total time spent traveling on IT_{U_i}:

$$TD^{*}_{U_i} = \sum_{w=1}^{n} Min_{V_j \in V_w} \left(TD^{j}_{PI_{w,i}}\right) = \sum_{w=1}^{n} Min_{V_j \in V_w} \left(w_{1_i} W_{j,w} + w_{2_i} T_{j,w}\right) \quad [1.21]$$

1.3.6. *Geographical network subdivision served and implementation of a physical distributed dynamic architecture*

In order to alleviate the impact of heavy processes on the quality of service provided and response time, we propose to develop a platform that promotes the decomposition of the initial overall process into several less complex tasks. These tasks will be carried out in a decentralized manner and in parallel to reduce the complexity of processes performed.

To do this, a status representation of the current geographic network served is established as a distributed dynamic graph, taking inspiration from some existing works using the same principle [FEK 10; KAM 07; HIZ 08].

a) Decomposition areas

For the decomposition of the network served, considering an effective method to encompass all available data while avoiding duplication of information overrides any other objective. We propose to achieve this by a principle of subdivision to create a set of distinct geographical areas whose main features are:

– All areas created represent geographical areas of similar shapes and sizes.

– Each area has its own perimeter that encompasses an information set relating to requests and/or offers of a car-sharer. Thus, each zone corresponds to a software component, more or less independent, locally processing the data included in each area. A set of autonomous (or semi-autonomous) components is created for established areas. They run in parallel so as to optimize the overall system processing.

– Two or more zones may have common areas of intersection, representing a single point or several points as they may represent an empty geographical area.

– Two or more zones can intersect and share data which could help proper implementation of the system; the data redundancy within zones is useful because each component created needs comprehensive knowledge of the system in order to best address the user requests

located in specific areas. However, it can conflict when two or more areas share data on the origins of user requests.

– In what follows, we focus particularly on methods for the classification of data according to one or more matching criteria for the creation of scatter points. The idea behind these methods, and especially in "*unsupervised classification*", is to create a set of k classes, where each class corresponds to a set of individuals. Areas of high density are then responsible for creating "clusters" with which class labels are associated. Observations on the network are thus analyzed and studied based on well-defined criteria, in this case distance, in order to later be able to collect a set of neighboring classes in space.

– The network served that we model is reduced to a two-dimensional space where the information that we have is presented as a homogeneous dataset. These data relate to geographical points represented by their GPS coordinates (i.e. longitude and latitude). Taken together, these data, which constitute the observations of the system, can be classified in such a way that the dominant character of separation is the distance between each two nodes of a population belonging to the same cluster. A real underlying class hence occupies a limited time–space region of the map representing the network area served. Each class is formed relative to a set of nearby nodes observed by the previously detailed graphical modeling, which may be decomposed and thus give rise to a multitude of clusters.

– As we have previously observed, two or more zones can overlap and share one or more individuals (i.e. nodes) with different degrees of membership and whose sum is equal to one [AMM 07]. Based on this principle, we are then placed in the context of an unsupervised blurry classification [ALL 06] where the number of groups is unknown and must be defined by the decomposition algorithm. Also, being placed in the context of a dynamic car-pooling service where supply and demand can occur at any time, it is imperative to consider the evolution of the number of individuals for continuous updating of classes thus established. The number of clusters may change during the evolution of processes and hence throughout the allocation process or the decomposition process.

– Each zone is associated with a region of "*similar*" observations according to the established criteria (i.e. close in space according to an initial pre-determined limiting distance). Based on what has preceded it, an area contains the following parameters:

- cardinal (N): the number of individuals;

- center (Ci): the center of area Zi;

- radius (r_i): the radius of the area Z_i . This radius is the same for all established areas.

– From the set of determined neighbors at each iteration of the subdivision algorithm detailed later, the properties of each class are created and hence defined, essentially its center and radius. From these parameters, new populations can be created determining the breadth of knowledge for the identified areas ("classes") [ALA 06]. The center of the area is calculated so that it is the equidistant point (i.e. center of gravity) between the furthest points of the set being considered. Several scenarios can occur and should be considered as long as the set of determined neighbors contains one, two, three or more nodes:

- the determined set contains a single node removed from the other points in the network at a strictly greater distance than the predetermined diameter of the system;

- if the neighboring points are two in number and no node belongs to the perimeter of the area including that pair of nodes, the center of the area that includes them is calculated as the midpoint of the line connecting the two points in question;

- the area in question is subsequently established based on the equidistant point calculated from the two given nodes and its, in theory, pristine neighborhood. However, the knowledge space of an area may evolve to include other nodes related to new offers or requests for car-pooling;

- if the set of neighbors has a cardinal of three, then the center of the area that contains all of these nodes is the center of gravity of the triangle whose vertices are represented by points of this set;

- if the set of neighbors is of size three or more (Cardinal (N)≥3), then the center of the area that contains all of these nodes is the center

of gravity of the triangle whose vertices are represented by points of this set. In fact, it is the center of gravity identified as the isobarycenter of these three vertices and the center of mass of the interior of the triangle. This point is located two-thirds of the way along from each center starting from the vertex.

b) Decomposition principle

Step 1: Creation of Primary Zones

Algorithm 1.1 describes the software steps followed by our system in order to create the Primary Zone. These steps rather refer to a set of iterations during which the sets of neighbors are defined and their respective centroids are identified, on the basis of what the areas are established on.

Algorithm 1.1. Creation of Primary Zones (CPZ)

Data: $SN = N_R(t)$: set of origins and destinations nodes of requests, Diameter

Result: PZ: Set of Primary Zones

Initialization: Number k = 0, Distance = 0, VN: Set of nodes visited = ϕ, N: Table of set of neighbors = ϕ

While $VN \neq SN$ **Do**

Random selection of a node n of $\{SN - VN\}$

$N[k] \leftarrow N[k] \bigcup n$

$VN \leftarrow VN \bigcup n$

For All node n_1 **such that** $n_1 \in \{SN - VN\}$ **Do**

For All node n_2 **such that** $n_2 \in N[k]$ **Do**

$Distance = Distance\ (n_1, n_2)$

If $Distance \leq Diameter$ **Then**

$N[k] \leftarrow N[k] \bigcup n1$

$VN \leftarrow VN \bigcup n1$

End If

End For

End For

$k \leftarrow k + 1$

End While

For All number i **such that** $i \in \{0 \cdot\cdot k\}$ **Do**

$C_i \leftarrow Center(N[i])$

$PZ_i \leftarrow Establish\_Primary\_Zone(C_i, Diameter)$

$PZ \leftarrow PZ \bigcup PZ_i$

End For

Return PZ

Step 2: Subdivision of the network

In this second step of subdivision of the network, all information (*node*) not processed in the first step is taken into account. This information is considered with the aim of finding allocations for areas already established or creating new ones based on the same neighborhood principle.

These areas are said to be secondary areas or Intermediate Zones (*ZI*), since they only enter the allocation process if it is necessary. The overall set Z includes all the areas created, whether primary or secondary, that are likely to be questioned, updated or perhaps even eliminated, depending on the flow of information received. This is a view of optimal processing, avoiding duplication of information and/or processing, while covering the search space in question.

Algorithm 1.2. Creation of Intermediate Zones (CIZ)

Data: $SN = N_O(t) \bigcup N_{ID}(t)$: set of origin, destination and intersection point nodes of vehicles, number k

Result: IZ: Set of Intermediate Zones

Initialization: Number m = k, Distance = 0, Diameter, VN: List of nodes visited = ϕ, N: List of neighbors = ϕ, Boolaen*Allocated = False*

While $VN \neq SN$ **Do**

Random selection of a node *n* of $\{SN - VN\}$

$VN \leftarrow VN \bigcup n$

For All number *j* **such that** $j \in \{0 \cdot\cdot m\}$ **Do**

If $j \leq k$ Then

$C_j \leftarrow Center(Z_j)$

Distance $\leftarrow$ Distance(n, C_j)

Otherwise

Distance $\leftarrow$ Greatest$(N[j])$

End If

If $Distance \le Diameter$

Then

$$N[j] \leftarrow N[j] \bigcup n$$

$$\text{Allocated} \leftarrow \text{True}$$

End If

End For

If Allocated = False **Then**

$$m \leftarrow m + 1$$

$$N[m] \leftarrow N[m] \bigcup n$$

End If

End While

For All i **such that** $i \in \{k \cdot\cdot m\}$ **Do**

$$C_i \leftarrow Center\left(n[i]\right)$$

$$IZ_i \leftarrow Establish\_Intermediate\_Zone(C_i, Diameter)$$

$$IZ \leftarrow IZ \bigcup IZ_i$$

End For

Return IZ

1.4. ODCCA: an optimized dynamic car-pooling platform based on communicating agents

The necessity for the establishment of an effective process that combines the proposed "*physical*" architecture of the land is all the more imminent as it contributes to the effectiveness of processing, while following the evolution of the image of the car-pooling network. This contribution lies with the fact that establishing a decentralized and relocated request optimization process of several software entities works in parallel. These entities that we place in a multi-agent system context are also assimilated to *"distributed"* software agents, similar to the graphical modeling proposed in the previous section. Their main role is to carry out an optimized and efficient allocation of system data in the way that they are distributed in the geographical network served.

1.4.1. *Multi-agent concept for a distributed car-pooling system*

The modeling of a geographical network as a Distributed Dynamic Graph is conducive to the deployment of a software architecture composed of a multitude of autonomous entities operating in parallel and promoting the reduction of processes by reducing the complexity of the problem. In what follows, we propose to establish a multi-agent system called ODCCA, which stands for Optimized Dynamic Car-pooling based on Communicating Agents, which is provided with a multitude of features.

Functional analysis of the dynamic car-pooling problem

In what follows, five major steps (Figure 1.2) are required with the ultimate aim of ensuring a high quality service with a maximum rate of positive responses. The idea encompasses different preliminary steps preceding the application of algorithms for the allocation of vehicles to users.

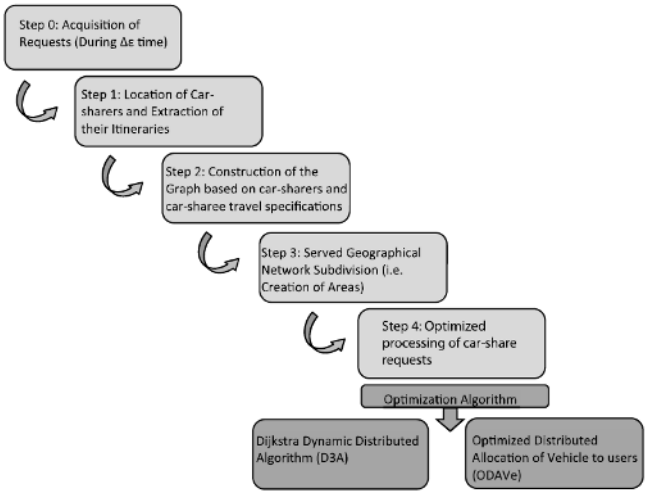

Figure1.2. *Features of the proposed system*

Step 1: Acquisition of requests

This involves the ability to simultaneously acquire specifications for car-sharers issuing simultaneous or near-simultaneous requests. A new parameter Δε has been introduced; it consists of a negligible lag time during which the system will wait for any car-pooling requests before starting the allocation search process.

Two pivotal functions govern the parallel processing of data on pedestrians and their travel requirements.

The first relates to the acquisition of requests issued during the time lag indicated (i.e. Δε).

The second relates to the organization of the set of requests received in order to highlight similarities and promote parallel processing (Figure 1.3).

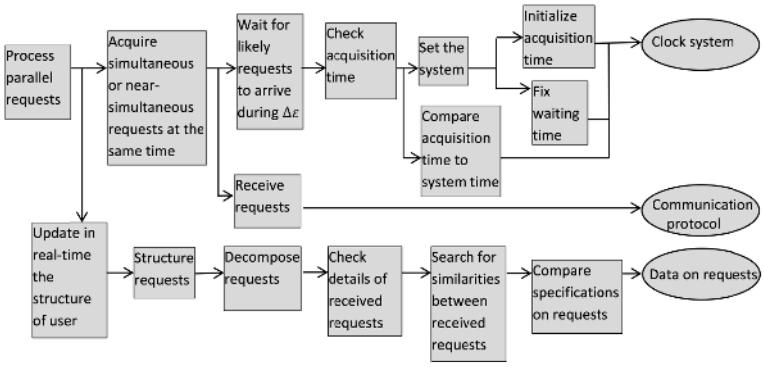

Figure 1.3. *Reception and structure of quasi-simultaneous requests*

Step 2: Extraction of car-pooling offers

The proposed system must process real-time offers of journeys by continually updating the search space to include the new features of the offer in question. A continuous and dynamic update aims to integrate any newly received offer in order to consider it in the allocation process and perhaps offer it as a possible solution, if it matches.

Figure 1.4 illustrates the operation of the system for the identification of car-pooling offers.

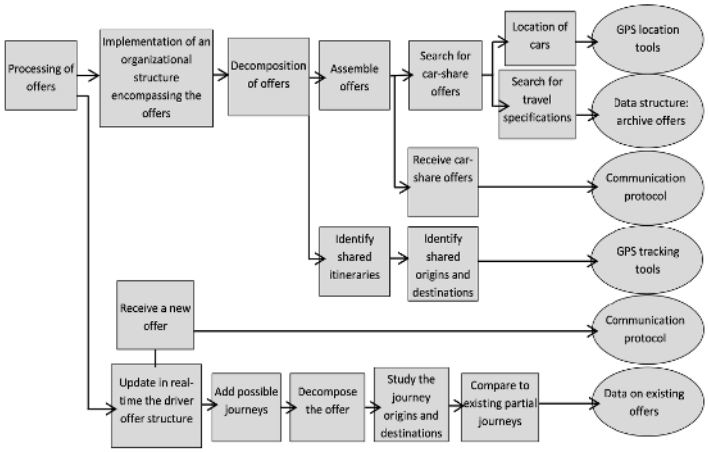

Figure 1.4. *Extraction and organization of car-pooling offers*

Step 3: Construction of the graph

From the two steps above, the matrix representations have resulted in the emergence of the concept of graphics based on graph theory in order to illustrate a real view of the served network. This step involves the establishment of a fusion graph where a superimposition of two planes in space is performed; the first represents the requests of pedestrians and the second represents driver offers. A continuous real-time update of the graph must be established and eventual modifications may occur when an external event occurs. The FAST diagram shown in Figure 1.5 shows the process of deploying different system input data in graph form.

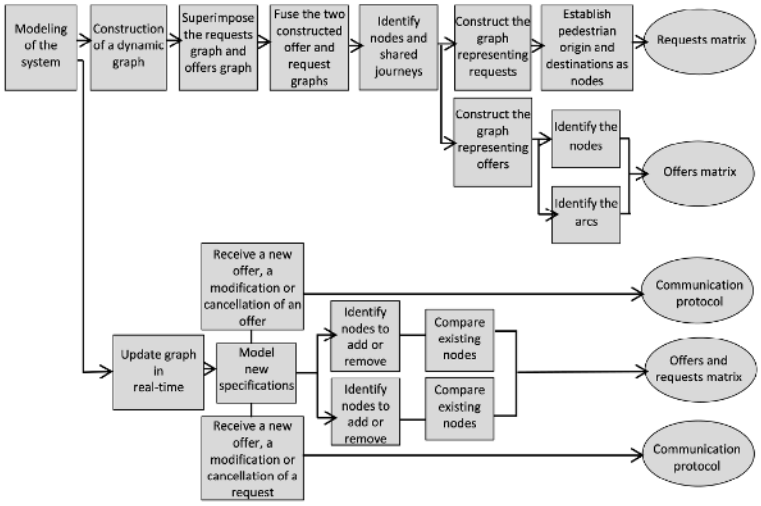

Figure 1.5. *Construction of a representative model of the system*

Step 4: "Divide and conquer"

In this section, we propose subdividing the geographical network served to highlight areas of greater or lesser density. Consequently, the process of finding optimized allocations will be decomposed and relocated in the extracted areas and is thus led to the image of the network after subdivision.

Figure 1.6 shows the result of the decomposition principle stated above.

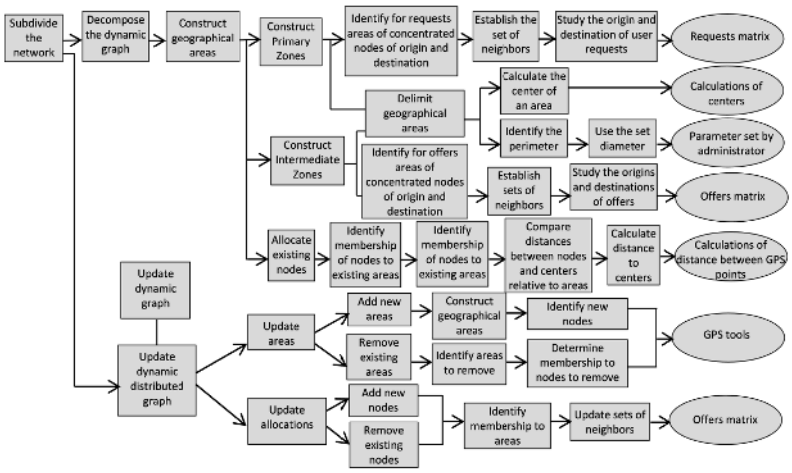

Figure 1.6. *Decomposition of the network*

ODCCA: A multi-agent architecture

Given the exponential complexity of the problem of optimized dynamic car-pooling, a decentralization and division of tasks into a set of software components is required. These components are presented as autonomous intelligent entities running in parallel in a rapidly changing and unstable environment. A multi-agent architecture is therefore proposed in Figure 1.7.

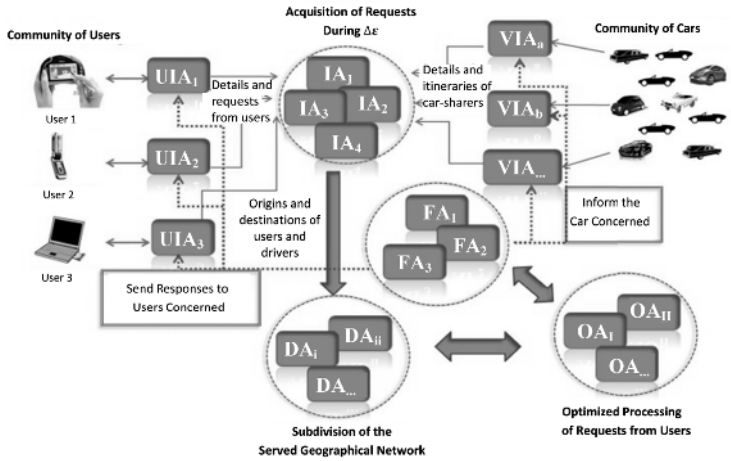

Figure 1.7. *ODCCA: a system based on Communicating Agents*

Each agent has its own action space and features. In what follows, we present the different agents currently in place, as well as details on their operation and their contributions to the optimized allocation process of vehicles to users.

– User Interface Agent (UIA): immediately from their connection to the system, any user of the service is supported by a UIA. The features of a UIA are summarized as receiving a request for car-pooling, checking the coordination and defining the default values specified by the user.

– Vehicle Interface Agent (VIA): this is to provide, at the core of the system, an interface through which it can communicate with drivers of cars. In parallel to communication with pedestrians, potential passengers, the system must be able to interact with car-sharers. These exchanges will be at the service of the coordination between drivers and passengers to be able to organize their journeys and automatically synchronize pick-ups and drop-offs. A (VIA) agent will also collect the necessary information about their offers and check the settings to transmit information directly to the information agent (IA) if the process is already underway.

– Information Agent (IA): the function of this agent is to collect data necessary for the proper functioning of the system, bearing in mind that any existing information may contribute to the description of the state of the environment and organize them in a suitable data structure.

– Decomposing Agent (DA): this agent is responsible for the process of subdividing the served network. The pre-defined decomposition algorithms are driven by the agent for the establishing sets of neighboring nodes formed on the basis of data received from the IA on offered and requested routes. The DA extracts "*communities*" of nodes in proximity. These help define zones, whether they are primary or secondary, and their geographical stretch due to the calculation procedure of the centers and the preset diameter.

– Optimizing Agent (OA): two types of optimizing agent are implemented in our system. Each type depends on a well-defined optimization algorithm which is located in the core of the corresponding agent. Moreover, both share optimization tasks on different instances initiated by the corresponding type. They perform

their tasks locally in the area for which they are responsible then come together in a coalition in order to reconstruct complete and consistent solutions. The bodies in question, whether they belong to one or the other of these two types, converge in a way that their behaviors fall within the context of distributed optimization.

– **Fusion Agent (FA)**: this agent is instantiated as soon as a set of solutions is generated by the system. It is responsible for identifying the user involved in each solution received, identifying the nature of the response (empty or invalid) and acting accordingly. It is also equipped with the ability to be the middle agent between the UIA and the VIA to validate, or not, a trip for the users concerned.

Entities established in our system must allow full synchronization without conflict and maintain an ambient intelligence throughout the system. For this, the establishment of communication channels between the agents is required, which is where the selection of an appropriate modeling language for this context, the UML, comes from.

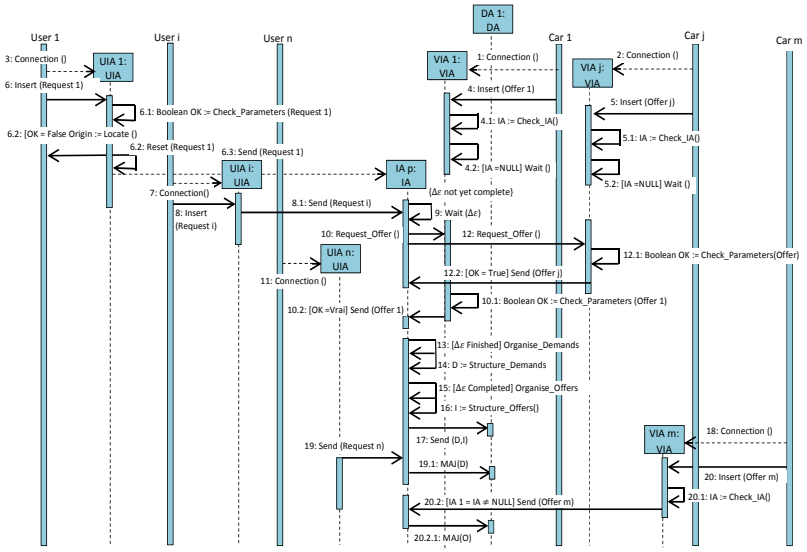

Figure 1.8. *Collection and organization of data on the supply and demand of car-pooling*

Figure 1.8 shows the collection and organization of data on car-pooling supply and demand. The purpose of this coalition is to gather information and to structure it so that they can be manipulated by the DA and OA, and thus pave the way for the allocation processing of vehicles to users.

1.5. Formal modeling: for an optimized and efficient allocation method

After extraction from space of possible solutions, the proposed system goes on to the second stage of its operation; the search for the best solution, given the optimization criteria set here, is established as the overall duration of the journey. A different approach is proposed in each case based on two principles of adopted resolution. The first is an adaptation of Dijkstra's algorithms and the second consists of an exact optimization method. Both approaches are detailed in this section.

1.5.1. *D3A: Dijkstra Dynamic Distributed Algorithm*

In this first approach, we are interested in finding optimized itineraries based on the principle of Dijkstra distributed algorithms and to distribute the execution primarily on *DA* and *OA*. This process corresponds to the execution of Algorithm 1.3 provided in the following:

Algorithm 1.3. Dijkstra Dynamic Distributed Algorithm (D3A)

Data : $G = (N, A)$: Graph modeling the network, R : Requests by users,

Result : OS : Optimized Solutions

Initialization :

Node $D \leftarrow NULL$

For All node n **such that** $n \in N$ **Do**

$Weight[n] \leftarrow \infty$

End For

For All request r_i **such that** $r_i \in R$ **Do**

$Weight\left[r_i^+\right] \leftarrow d_i$

End For

For All OA_i **Do** {only the OA_i on the PA containing sources are launched}

$Weight_{OA_i}, Car_{OA_i} \leftarrow A3D(G_i, TPM, LV, R_i)$ {Launch algorithm *D3A* on each OA_i }

End For

For All request r_i **such that** $r^i \in R$ **Do**

$D \leftarrow r_i^-$

While $\left(D \neq r_i^-\right)$ **Do**

Insert D at the start of SO_i

$Z_i \leftarrow Locate(D)$ {Determine the zone that contains D to consider results provided by *OA* concerned}

$OS_i\left[\Pr edecessor\left[D\right], D\right] \leftarrow Car_{OA_i}\left[\Pr edecessor\left[D\right], D\right]$

$Pl_{Car\left[\Pr edecessor\left[D\right], D\right]} \leftarrow Pl_{Car\left[\Pr edecessor\left[D\right], D\right]} - P_i$

$D \leftarrow \Pr edecessor\left[D\right]$

End while

End For

$OS \leftarrow OS \bigcup OS_i$

Send *SO* to *FA*

Algorithm 1.4 provides the detail of application of the Dijkstra algorithm locally to each zone by the responsible OA. This algorithm is performed in order to look for possible allocations, particularly those that optimize the waiting and travel times, thus promoting earliest arrival at the desired destination.

Algorithm 1.4: Dijkstra algorithm optimized for the allocation of vehicles

Data: $G_i = (N_i, A_i)$: Graph modeling of zone Z_i , TPM (average travel time function), R_i : Requests whose origins are in Z_i , LV: List of Vehicles.

Result: *Weight*: List of node weights, *Car*: List of cars affected by sections of itineraries

Initialization:

For All node n **such that** $n \in N_i$ **Do**

$Weight[n] \leftarrow \infty$

End For

For All request r_i **such that** $r_i \in R_i$ **Do**

$Weight\left[r_i^+\right] \leftarrow d_i$

End For

$Q \leftarrow N_i$

While $Q \neq \phi$ **Do**

$p \leftarrow \min(weight)$ {choose the node with smallest weight}

$Q \neq Q - \{p\}$

For All node q **such that** $(p,q) \in A_i$ **Do**{nodes of the set of

neighbors of p }

For All $o_j \in O$ **Do**

If$\left(Check\_Constra\text{int}s(o_j, r_i)\right)$ **Then**

$W_t \leftarrow w_1 * Dis\tan ce\left(o_j^+, p\right) * TPM\left(o_j^+, p, d_j\right)$

If $(W_t + t - Weight[p] \succ 0)$ **Then**

$W_{tj} \leftarrow W_t$

End If

End If

End For

$c \leftarrow \min(W_t)$ {Choose car with shortest waiting time}

$a \leftarrow Weight[p] + W_{t_c} + w_2 * Dis\tan ce(p,q) * TPM\left(p, q, Weight[p] + I\right.$

If $a \prec Weight[q]$ Then

$Weight[q] \leftarrow a$

$\Pr edecessor[q] \leftarrow p$

$Car[p,q] \leftarrow c$

End If

End For

End while

Send *Weight* to adjacent areas

Return *Weight* and*Car*

1.5.2. *ODAVe: Optimized Distributed Allocation of Vehicle to users*

In order to improve system performance, this second approach has been suggested with the aim of supporting an optimized "*if possible*" allocation search principle, while focusing on collaborative work between different *OAs*.

Indeed, as shown in Figure 1.9, the idea is to find the nearest node served, decompose the request, solve the first stage of the request, send a message of type "Request/Response" to the OA responsible for the zone containing the nearest node and then repeat on the next closest set of nodes served until the solution is non-empty or until there are no more nodes to visit.

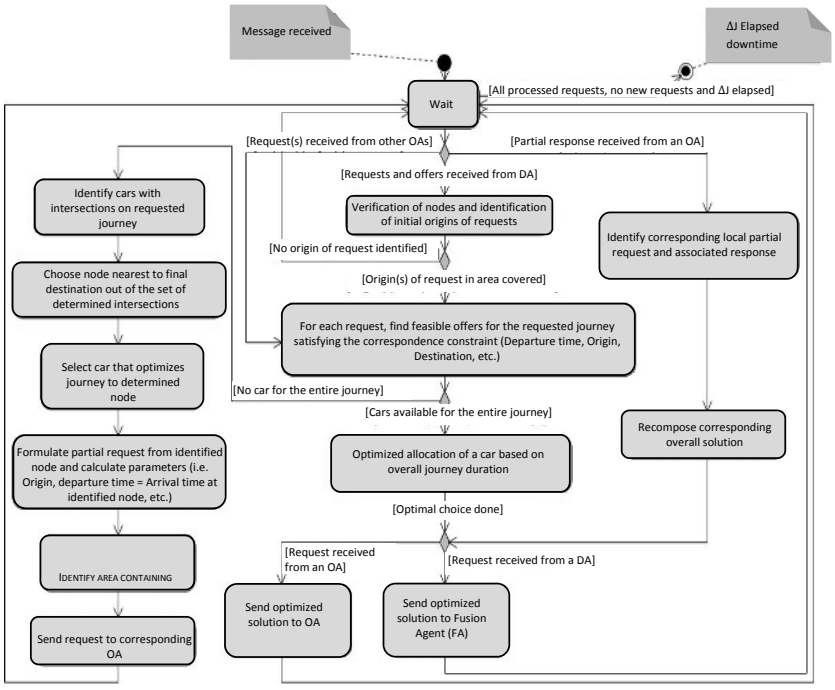

Figure 1.9. *An OA for the optimized distributed processing of requests based on ODAVe*

All operations with the appropriate modeling are detailed throughout algorithms 1.5 and 1.6 below.

Algorithm 1.5. Optimized Distributed Allocation of Vehicle to users (ODAVe)

Data: R_{AO_p} : Requests whose origins are located in zone A_p incumbent upon OA_p; O(t) I(t): Set of offers, TDT : Tolerable Delay Threshold, AIS: Average Itinerary Speed, t : system time, $\{\omega_1, \omega_2\}_R$

Result: S : Set of solutions for R

Initialization: Solution $S \leftarrow \phi$

For All request r such that $r \in R_i$ Do

$$S_i \leftarrow OGI\left(r_i\right)$$

If $S_i = \phi$ Then

$$IN \leftarrow IN \bigcup Nearest\_Node\left(I\left(t\right), r_i^-\right)$$

Repeat

$$NP \leftarrow Nearest\_Node\left(IN, r_i^-\right)$$

$$IN \leftarrow IN - NP$$

$$r_{i_l} \leftarrow Establish\_\mathrm{Re}quest\left(r_i^+, NP, P_i, d_i, a_{NP}\right)$$

$$S_{i1} \leftarrow OGI\left(r_{i1}\right)$$

$$Z_c \leftarrow Area\_Containing\left(NP, r_i^-, P_i, d_{NP}, a_{r_i-}\right)$$

$r_{i_2} \leftarrow Establish\_\text{Re}quest\left(NP, r_i^-, P_i, d_{NP}, a_{r_i-}\right)$

Send r_{i_2} to $OA_{c\{}OA_c$ is the OA responsiblefor $Z_{c\}}$

$S_{i2} \leftarrow ODAVe_{AO_c}\left(r_{i2}\right)$

IF $S_{i2} = \phi$ Then

$S_{i2} = \phi$

Otherwise

$S_i = S_{i1} \oplus_{i2}$

End If

Until $S_i \neq \phi$ Or $IN \neq \phi$

End If

$S \leftarrow S \bigcup S_i$

End For

Send S to AF

Algorithm 1.6 below is relative to the function searching for an offer that may respond to the desired path in its entirety.

One option called PS (*Possible Solution*) in our algorithms is thus considered:

$$SP = \left(Departure\_Po\text{int}, Arrival\_Po\text{int}, V_j, Pl_j, Duration, d, a, Fitness\right).$$

Algorithm 1.6. Overall Optimized Itinerary (OOI)

Data: R_i : A specific request included in zone Z_p incumbent upon OA_p; $\mathrm{O(t)}$: Set of offers, t : system time, $\{w_1, w_2\}_R$

Result: S_i : Overall optimized solution for R_i

Initialization: Solution $S_i \leftarrow \phi$, $SP \leftarrow \phi$: set of possible solutions

For All offer o_j such that $o_j \in O(t)$ Do

If $R_i^+ \in IT_{v_j}(t)$ Then

If $R_i^- \in IT_{v_j}(t)$ Then

Calculate $(W_t, T_t, Duration, d, a, Fitness)$

If $\left(Check\_Constra\,\mathrm{int}\,s\left(d_i, Pl_j, d, a\right)\right)$ Then

$SP \leftarrow SP \bigcup \left(R_i^+, R_i^-, V_j, Duration, d, a, Fitness\right)$

End If

End If

End If

End For

$S_i \leftarrow Choose\ SP\ of\ SP\ that\ optimizes\ Fitness\ Function$

Return S_i

1.6. Implementation and deployment of a dynamic car-pooling service

In this phase, it is necessary to specify how the intervening agents will be deployed in the overall process of our approach. The distribution of the agents is of the utmost importance as it directly affects the cost of communications, monopolization of resources and optimization of resource consumption, etc. It implies specifying the correct way to carry out their deployment in order to optimize the use

of resources and save time. The direct impact on the costs in terms of communication stems from the fact that the distribution of agents can promote their optimization or conversely, degrade their quality. Besides the fact that the sending and receiving of a message depends primarily on the type of network in which agents are deployed, it is also necessary to optimize (time) costs of transmission of messages between agents by finding good distribution thereof.

Our approach, as we have defined it, inherits characteristics from the different agents that make it up. In the main optimization algorithms at the heart of the two processes following which we approach the issue of CDOs, we can note that an agent is only created and only intervenes if a need is expressed. As such, we can make our strategy only involve UIAs that relate to users who issued car-pooling requests in our platform. Also, in this context, the principle of decomposition of the way in which it was approached joins with the idea of an "*economy of agents*" by creating OAs only on PZs that were accomplished at the very beginning of the process. An *IZ*, conversely, would only come into play if necessary, and the responsible *OA* is only created if the decomposition of a request requires this (when the intermediate point joining the two partial requests can be found in the IZ concerned). Subsequently, in addition to the benefits of the concept agent, the adopted resolution methodology, whatever the relation to the principle of decomposition or to the distributed optimization methods, brings a significant advantage in optimizing communication costs.

Furthermore, independent of the network topology as well as its nature, the capacity of communication that the agents in our system are equipped with offers the possibility of adaptation and a wide variety of deployment configurations.

In what follows, we present the architecture adopted for our system, and the implementation of the system (ODCCA) which shows different scenarios for the platform execution thus established, called *DOMARTiC* (Distributed Optimized approach based on the Multi-Agent concept to set up a Real-Time Car-pooling service). Moreover, in each test carried out, a representation of the geographical network

served was performed thanks to real mapping with integrated software CartoCom.

For the implementation of our ODCCA system or DOMARTiC, the JADE (Java Agent DEvelopment framework) platform [BEL 03; BEL 99] was chosen as the development platform, allowing for more compatibility between agents, total interoperability, and a secure exchange of communication.

1.6.1. *Deployment of ODCCA: choosing a hybrid architecture*

In a centralized architecture, all agents can be found on a central server to which all car-pooling requests and offers are redirected. This configuration has the advantage of considerably reducing the costs incurred by the communication between the agents of our system in the same place and hence avoiding a pointless exchange with a distant network. However, this configuration has the disadvantage of making the agents work sequentially, despite the fact that they are supposed to work in parallel to save time and therefore improve performance. On the other hand, the distributed architecture also has major advantages of which the main objective concerns the optimization of the execution time with the launch of several processes in parallel for the treatment of different processes, including, in our case, the dynamic optimized allocation. These characteristics represent a considerable improvement in the execution time of our system, which allows us to stay in the field of approach dynamics and achieve high performance in real-time systems.

To do this, and thus take advantage of one or the other of the two approaches, we jointly consider both configurations centralized and distributed in a single "*hybrid architecture*". As can be seen in Figure 1.10, the deployment of our request involves several stations to be interconnected through a local network, thus avoiding wasting the time gained by the introduction of parallel processes in long exchanges between agents dispatched on a larger network. In addition to the parallel processing and exchange in local network that supports the optimization of the execution time and thereby increase system performance, we also limit the physical resources used. For example,

a group of three machines (Figure 1.10) shows greater performance than if it were bigger because they each host multiple agents, and the exchanges thus done internally positively affect the performance of the request thus deployed.

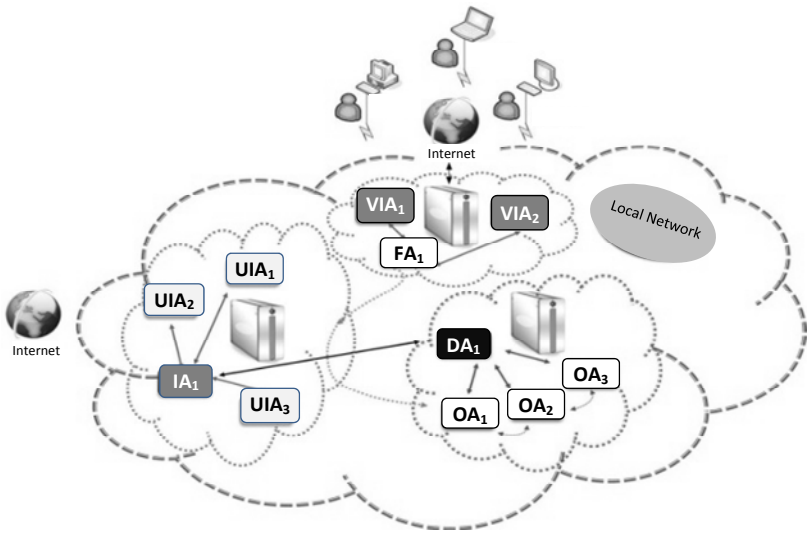

Figure 1.10. *Deployment of our ODCCA system*

In this architecture, an example of deployment of instantiated agents in the context of the completion of ODCCA shows dispatching on three different machines connected through a local network, each with a quad-core processor, a principle and technology that we use to optimize flow while reducing the cost.

Based on this architecture and the development platform, the software architecture is well defined and fulfills a separation between presentation layers, processing and data while setting up different forms of interaction between them. To demonstrate the composition, we present the MVC model that meets the requirements of our system.

1.6.2. *Layered architecture*

In this section, we present the software architecture that represents its composition components or entities involved in the overall operation. We propose to set up our system to adopt the *MVC* architectural model, the basic features of which are explained in the next section.

– MVC model

Our system is modeled as a layered architecture involving three different implementation layers of our system, namely a Model layer, a View layer and Controller layer. More generally called *MVC* (*Model–View–Controller*), it is a design pattern designated rather as an architectural design based on the separation of data (the *Model*), the Human–Machine Interface (the *View: HMI*) and control logic (the *Controller*). The three layers are distinguished as follows.

a) The Model

Specific to the data of a request, this represents the structure but also defines the interactions with the database and the processing thereof.

In our request, we specifically defined the characteristics of each of the agents involved in the implementation of the dynamic optimized allocation method. It should be noted that in the context of *MVC*, we can notice that the *Information Agent* (*IA*) must collect all the updated data required for processing to be performed. Thus it ensures its role to represent all data related to requests and offers of car-share.

b) The View

It covers everything related to interface with the user (*HMI: Human Machine Interface*), meaning that with which it interacts. No processing is done at this level except simply the display of data that it receives from the model layer. Multiple views can be initiated to represent the same data from a single model. In our case, for example, it is essentially views relative to specific interfaces for car-sharees (where the UIA is involved) and those relative to car-sharers (where the VIA is involved).

c) The Controller

This special layer is responsible for managing the interface and interaction between the *Model* and the client. It is therefore responsible for interpreting the request of the user in order to send the view to which it corresponds. The controller performs synchronization between the model and view layers and can thus generate events when the model changes and launch an update of the view already being displayed. In this context, we note that this is the *Fusion Agent* (*FA*) that provides information on all the changes generated on the model layer data (managed by the *IA*). The modifications, made by the *DA* and *OA*, will subsequently be dispatched to different instantiations of the *UIA* and *VIA* with a necessary identification of the corresponding user support so as to express the appropriate interface. In addition, the *DA* and *OA* also play a supervisory role over the views that we cannot ignore. They are actually responsible for the decomposition of the field in the first case and of the generation of optimized itinerary solutions in the second case. Different representations of the modifications generated on the graph that represents the network data thus emerge as a result of the implementation of these various stages of decomposition and search for allocations. A different view of the network can be displayed on a separate interface.

A three-layer architecture, Interface (Model), Application (Controller) and Data (Model), applied to our multi-agent model is shown in Figure 1.11. The latter shows a representation of the layers model with the model Agent proposed in ODDCA. This representation is consistent with the specifications of our problem, the chosen model and the principles and set and achieved treatments.

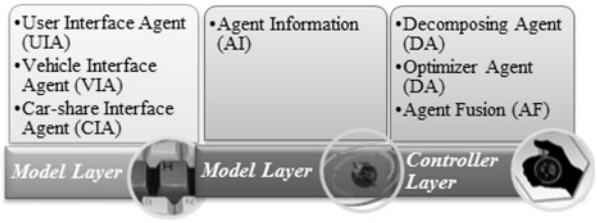

Figure 1.11. *System architecture and involvement of agents in each layer*

Figure 1.12 illustrates the model of the different flows showing exchange opportunities between the different layers.

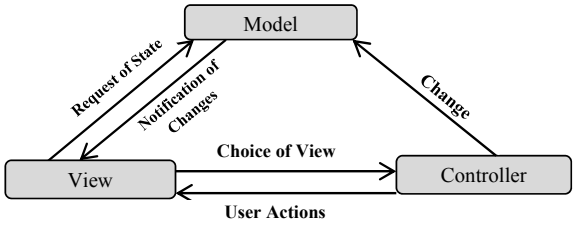

Figure 1.12. *Interactions between the layers of the MVC model*

By implementing our application following this architectural style, we benefit from its main advantages which are:

– the ability to change a layer without affecting the others since they are clearly separated and can be modified without affecting the rest of the layers in addition to the synchronization of the views.

– API JDBC

To access the data system, we made use of the *API* (*Application Programming Interface*) *JDBC* (*Java DataBase Connectivity*) that allows our request to access a database where all the information required for the current processing is stored.

JDBC technology is used to connect specific computer requests to a data server through the manipulation of existing data. It consists of a set of classes allowing the development of these types of request, providing them with the ability to connect to the DataBase Management Server (*DBMS*).

– CartoCom

The very imposing dynamic aspect in the context of total and continuous mobility with respect to the transport sector is very conducive to the allocation ofconflict traps that would no longer be in agreement when issued to car-sharees and car-sharers. Processing such evolving and inconsistent data from one moment to the next is a

choice that must be assumed in our system. It is within this framework that we have used the special services of a specific software package called CartoCom by the company Bayo[1].

Equipped with GPS devices, this platform facilitates the acquisition of geographic data and provides continuous tracking of users connected to the system. CartoCom has been ingrained into our platform as a *GIS* unit responsible for recovering data transmitted by users from their origins to their destinations by offering multiple services in parallel, which we took advantage of to achieve our goals. These mainly revolve around the desire to make our system full-scale; the mapping provided by the CartoCom software has enabled us to achieve, in a first instance, the desired objectives from this perspective and in a second instance, integrity and efficiency. Indeed, the services offered by this software are strongly conducive to "*correct*" manipulation free from conflicts, thus helping us to meet the real-time challenge. These services mainly include:

– The continuous geolocation of users of our system, whether they are pedestrians or drivers.

– The geolocation in frequency of scripts also allows full information on the movements of users for the creation of full traceability of the different routes taken by car-pooling or not.

– With CartoCom, our system favors the development of a continuous dialogue with users, taking advantage not only of the role of agents in this regard but of continually having essential information about their movements.

– Another service provided by CartoCom concerns the provision by the ODCCA system of real mapping on which users and their origins and destinations can be seen.

Figure 1.13 displays the solutions generated that represent the matching itineraries via CartoCom.

1http://www.bayo.com/.

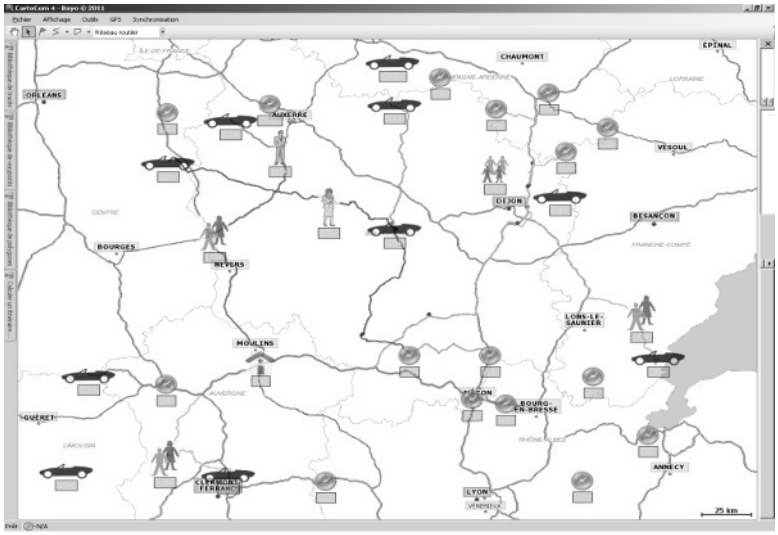

Figure 1.13. *Display of the solution itineraries. For a color version of the figure, see www.iste.co.uk/hammadi/multitransport.zip*

1.6.3. *Testing and implementation scenario*

To demonstrate the validity of the results generated through services provided by the DOMARTiC platform, implementation scenarios will be presented.

1.6.3.1. *User data*

Presented in Table 1.1 is the set of "*pedestrian*" users not yet allocated to a set of pseudo-simultaneous requests received at time *t=9:10*. It is the *IA* that ensured the acquisition. Having been created in the same instant as when the first user request was received, during waiting time $\Delta\varepsilon = 3$ *seconds*, the *IA* gathers information on the requests likely to occur in this time. It is assumed that there will be six and the specifications mainly related to the number of passengers (desired earliest departure and latest arrival time) are also presented.

User identifier	Request Origin	Request Destination	Number of passengers	d	a
U_1	U_1^+	U_1^-	1	9h10	10h40
U_2	U_2^+	U_2^-	2	9h10	10h
U_3	U_3^+	U_3^-	3	9h10	10h15
U_4	U_4^+	U_4^-	4	9h10	10h10
U_5	U_5^-	U_5^-	5	9h13	10h00
U_6	U_6^+	U_6^-	6	9h10	10h30

Table 1.1. *User requests at time t=9:10*

Conversely, Table 1.2 shows the different settings for vehicles emitting car-pooling offers.

Vehicle identifier	Offer Origin	Offer Destination	Number of passengers	Intermediate Destinations	d	a
C_1	C_1^+	C_1^-	1	$(C_5^+, C_8^-, U_3^-, C_7^-, C_5^-)$	-	10h15
C_2	C_2^+	C_2^-	3	$(U_4^+, ID_1, U_6^+, U_4^-)$	-	13h
C_3	C_3^+	C_3^-	4	(U_1^+, U_2^+, ID_1)	-	11h
C_4	C_4^+	C_4^-	3	(U_6^+, ID_1)	-	12h15
C_5	C_5^+	C_5^-	7	(C_8^-, U_3^-, C_7^-)	-	11h15
C_6	C_6^+	C_6^-	8	(U_5^+, U_6^+, U_5^-)	-	12h10
C_7	C_7^+	C_7^-	2	$(C_4^-, ID_1, U_1^-, U_2^-)$	-	12h
C_8	C_8^+	C_8^-	1	(U_1^+, U_3^+, C_5^+)	-	9h45
C_9	C_9^+	C_9^-	3	$(C_4^-, ID_1, U_1^-, U_2^-, C_7^-, U_6^+, U_5^-)$	-	13h15
C_{10}	C_{10}^+	C_{10}^-	2	-	-	13h

Table 1.2. *Car-pooling offers at time t=9:10*

Figure 1.14 shows the layout of nodes on a real map as generated by the CartoCom tool.

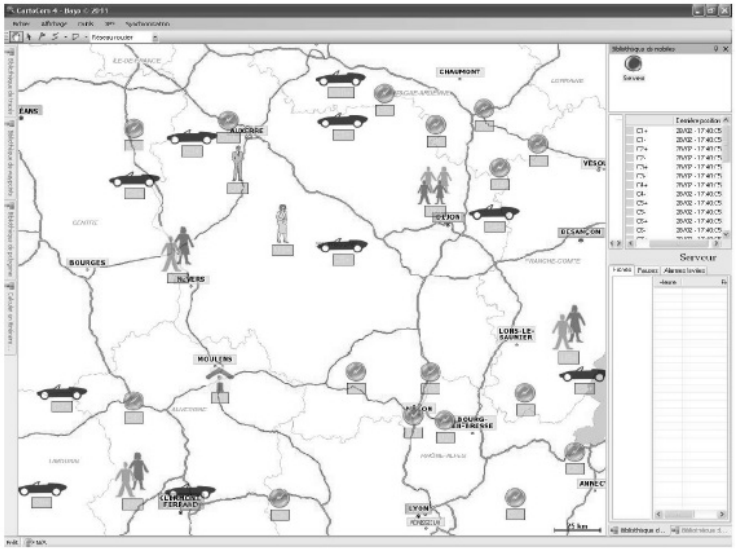

Figure 1.14. *Positions of the origins and destinations of pedestrians and vehicles on a real map. For a color version of the figure, see www.iste.co.uk/hammadi/multitransport.zip*

1.6.3.2. *Processing and allocation*

Processing really begins from the moment the various data are sent by the *IA* to the *DA* to undergo a first classification that highlights areas of data concentration, and where it should start the allocation process favoring those that are less dense and that do not contain request origins.

a) *Requests and primary areas*

The graph shown in Figure 1.15 relates to the modeling of pedestrian information (data in Table 1.1). In this graph, itineraries linking origins to desired destinations were constructed using the CartoCom tool, giving rise to the shortest journeys.

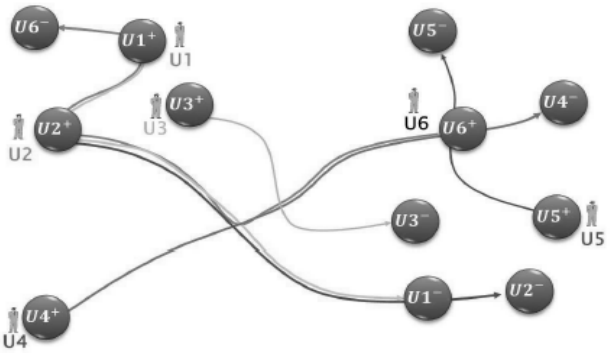

Figure 1.15. *Representation of users (pedestrians) and their itineraries. For a color version of the figure, see www.iste.co.uk/hammadi/multitransport.zip*

The Primary Zones (PZ) are established by the execution of algorithms named *CPZ* (*Creation of Primary Zones*). Firstly, the distances between the places of departure and arrival must be calculated. The distances calculated thus are used to determine membership or not of a set of nodes to the same area, compared to the determined diameter and for calculating the position of the centers of areas.

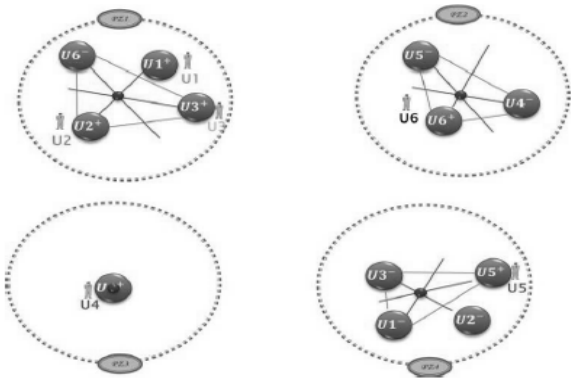

Figure 1.16. *Primary Zones*

b) ***Cars and intermediate zones***

As was done for car-pooling requests, almost the same procedure is followed in this second stage of decomposition.

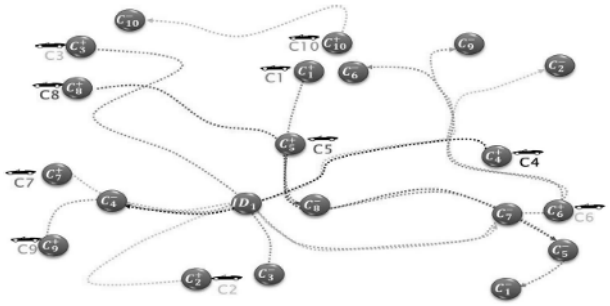

Figure 1.17. *Graph of origins and destinations of cars. For a color version of the figure, see www.iste.co.uk/hammadi/multitransport.zip*

Subsequently, a graph superposition of the two planes (pedestrian plane and car plane) in space is made creating a fusion of the two graphs to bring out the similarities and especially the correspondence between journeys.

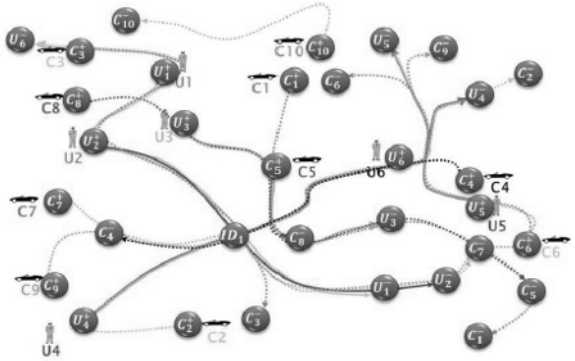

Figure 1.18. *Superposition of car-sharers and car-sharees. For a color version of the figure, see www.iste.co.uk/hammadi/multitransport.zip*

On the basis of this modeling and the Primary Zones created by the first phase of decomposition of the terrain (based on user requests only), the Intermediate Zones *IZ* will be gradually developed according to the *CIZ* (*Creation of Intermediate Zones*) algorithms previously developed and established. These first seek possible allocations of newly constructed nodes to previously developed *PZs*, and create new areas for those not allocated to the same neighborhood principle inspired by data classification methods (Figure 1.19).

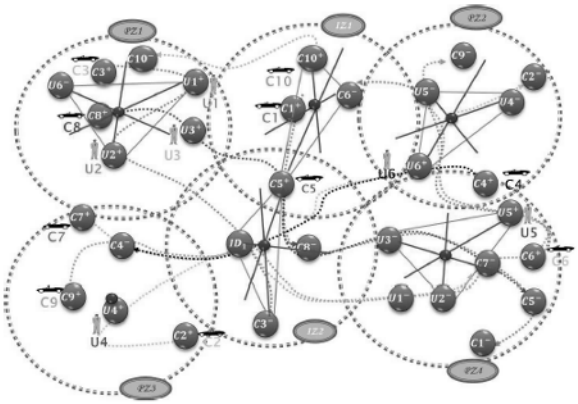

Figure 1.19. *Intermediate Zones and network coverage. For a color version of the figure, see www.iste.co.uk/hammadi/multitransport.zip*

1.6.3.3. *Some calculations*

Indeed, four *OAs* are created and launched at the same time to execute each *ODAVe* on requests whose origins lie in the geographical coverage area (PZ). Some examples, illustrative of calculations regarding intervening agents, are detailed in Figures 1.20–1.23 with the necessary data, particularly in relation to the distances between the nodes as well as the weights of the Fitness function specific to users affected by the request undergoing processing in each example.

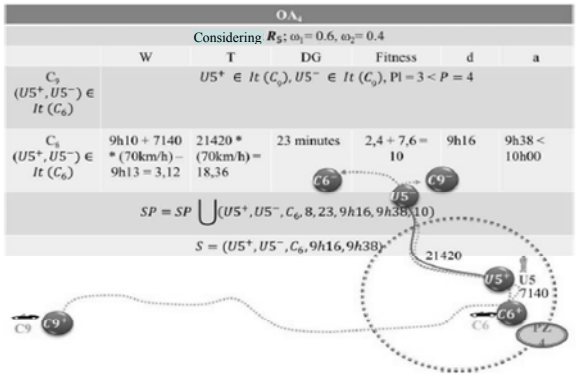

Figure 1.20. *Processing of R5. For a color version of the figure, see www.iste.co.uk/hammadi/multitransport.zip*

The R_5 example has two choices and coincidentally checks correspondence constraints. However, one of the two choices does not meet the feasibility of the offer.

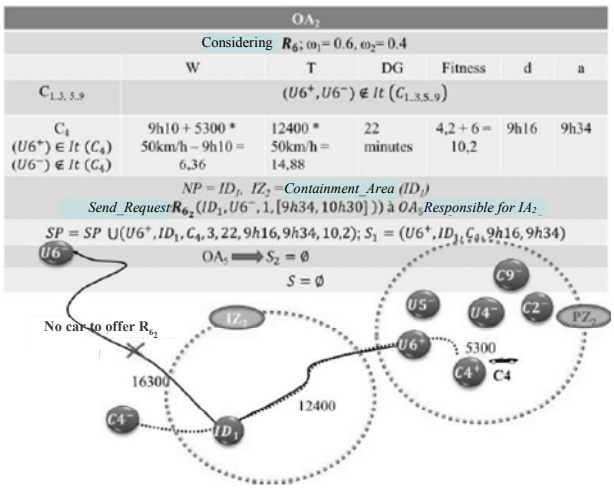

Figure 1.21. *Processing of R6*

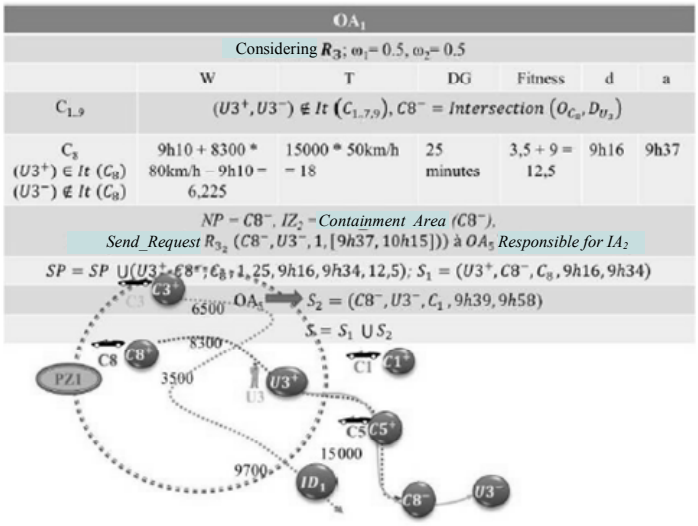

Figure 1.22. *Processing of R3. For a color version of the figure, see www.iste.co.uk/hammadi/multitransport.zip*

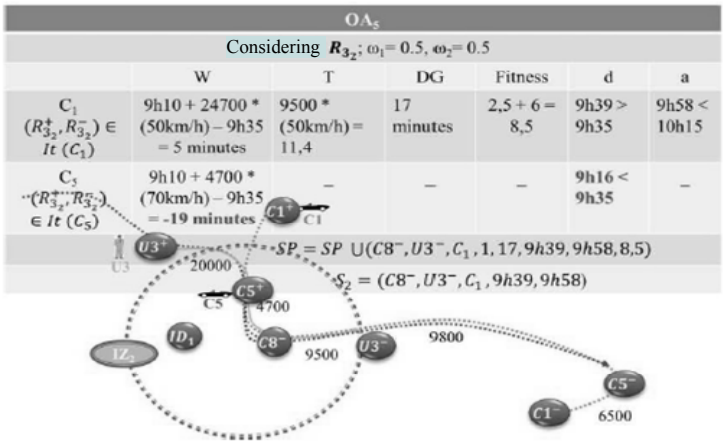

Figure 1.23. *Processing of the second part of R3. For a color version of the figure, see www.iste.co.uk/hammadi/multitransport.zip*

The last three figures highlight the collaborative operation of the agents for the treatment of R6 and R3 which involves a coalition between the OA2 and OA5 agents for the first part and OA1 and OA5 for the second part.

1.6.3.4. *Displaying the results*

Figure 1.24 displays the generated solutions that represent the corresponding itineraries via CartoCom.

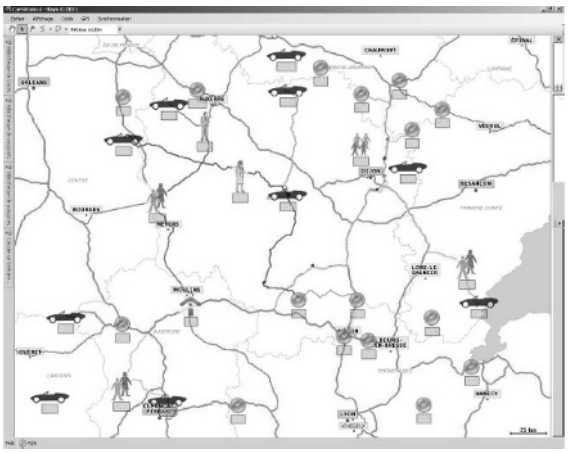

Figure 1.24. *Display of the solution itineraries. For a color version of the figure, see www.iste.co.uk/hammadi/multitransport.zip*

1.6.3.5. *DOMARTiC: the interface: execution example*

In what follows, we present different interfaces (Figures 1.26–1.27) generated by directly testing our application (DOMARTiC). A random generation module of "*mobiles*" (pedestrians and cars) has been developed to ease testing, all the while handling real geographic data. In Figure 1.27, we can see the different exchanges between the agents involved, especially in the context of a coalition.

Figure 1.25. *Random data generation*

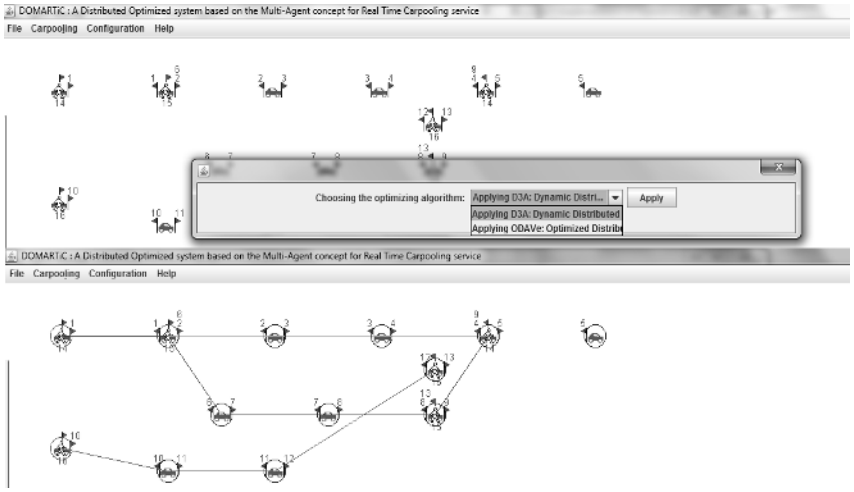

Figure 1.26. *Example of an execution. For a color version of the figure, see www.iste.co.uk/hammadi/multitransport.zip*

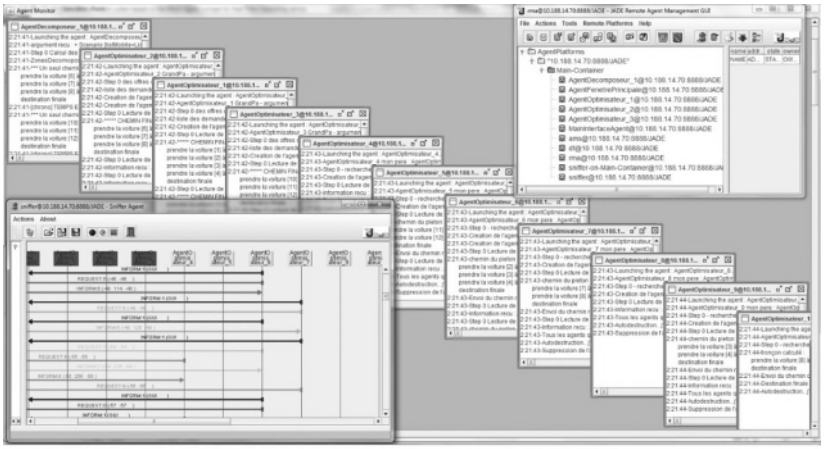

Figure 1.27. *Coalition between OA and recomposition of itineraries*

1.7. Conclusion

In this chapter, we presented the implementation of a dynamic distributed car-pooling system, ODCCA, for which we demonstrated the exponential complexity. We focused our efforts on the development of an effective resolution strategy leveraging a mixture of concepts, namely, multi-agent systems and optimization. A combination of these two founding concepts was the main contribution of this work since it exploits the benefits of both in a perfect and harmonious combination allowing a distributed artificial intelligence context to rule. Two approaches have been proposed in our system (ODCCA), the first proposal implements a Dijkstra distributed dynamic allocation (D3A) algorithm which was conducted to address the problems outlined previously and provide a complex service of polynomial order. Still in the context of multi-objective optimization, the second aspect concerns a heuristic combining even more optimization criteria. ODAVe implements a distributed optimization with the same goals as the first approach (i.e. to reduce complexity) by taking into account the lexicographical order on the set of optimization criteria (comfort and overall travel time).

To consolidate these choices in pursuit of a performance that will make ODCCA a competitive large-scale system, appropriate technical choices were required. To be noted in particular in this context: a JADE platform facilitating exchanges, a deployment model involving a hybrid architecture juggling between a centralized architecture and the distributed local network, an *MVC* model preventing any possibility of change or integration of new modules (i.e. statistics, planning and anticipation of travel, etc.) and a *GIS* module integrated via an interface with the CartoCom software. All these combined techniques and technologies lead the system, developed on this basis, towards efficiency.

1.8. Bibliography

[ALA 06] ALAMI J., EL IMRANI A., "Algorithme culturel parallèle pour l'optimisation multimodale", *Journal électronique d'intelligence artificielle*, June 2006.

[ALT 04] ALTMAN E., WYNTER L., "Equilibrium, games and pricing in transportation and telecommunication networks", *Networks and Spatial Economics: special issue on crossovers between transportationand telecommunication modelling*, 7-21, 2004.

[AMM 07] AMMOR O., RAISS N., SLAOUI K., "Détermination du nombre optimal de classes présentant un fort degré de chevauchement", *La revue MODULAD*, 37, 31-42, December 2007.

[ANI 92] ANILY S., HASSIN R., "The swapping problem", *Networks*, 22, 419-433, 1992.

[ATA 10] ATAHRAN A., LENTE C., T'KINDT V., "Transport à la demande et 'dynamic dial and ride problem': liens et état de l'art", *ROADEF 2010, 11e Congrès annuel de la société française de Recherche Opérationnelle et d'Aide à la Décision*, 2010.

[ASH 98] ASHOK K., Estimation and prediction of time dependent origin-destination flows, PhD Thesis, MIT, 1998.

[BAR 98] BARNHART C., JOHNSON E.L., NEMHAUSER G.L., SAVELSBERGH M.W.P., VANCE P.H., "Branch-and-price: 'column generation for solving huge integer programs'", *Operations Research*, 46, 316-329, 1998.

[BAR 07] BARNHART C., LAPORTE G. (eds), *Transportation*, Elsevier, Amsterdam, 2007.

[BEL 99] BELLIFEMINE F., POGGI A., RIMASSA G., "JADE – A FIPA-compliant agent framework", *Proceedings of the The Fourth International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents*, 1999.

[BEL 03] BELLIFEMINE F., CAIRE G., POGGI A., RIMA G., *JADE White Paper*, 3(3), 2003.

[BEN] BENOÎT R., *Communauto*, http: //www.communauto.com/.

[BER 10] BERBEGLIA G., CORDEAU J-F., LAPORTE G., "Dynamic pickup and delivery problems", *European Journal of Operational Research*, 202(1), 8-15, 2010.

[BOR 97] BORNDORFER R., GROTSCHEL M., KLOSTERMEINER F., KUTTNER C., "Telebus Berlin: vehicle routing scheduling in a dial a ride system", *Konrad Zuse Zentrum fur Information Technik Berlin*, 1997.

[BOU 02] BOULOGNE T., ALTMAN E., POURTALLIER O., "Mixed equilibrium in multiclass routing games", *Annals of Operation Research*, 109, 279-291, 2002.

[BRO 03] BROTCORNE L., LAPORTE G., SEMET F., "Ambulance location and relocation problem", *EJOR 147*, 451-463, 2003.

[CED 01] CEDER A., "Efficient timetabling and vehicle scheduling in public transport", in VOSS S. and DADUNA J.R. (eds), *Computer-Aided Scheduling of Public Transport*, 37-52, 2001.

[COR 02] CORDEAU J.-F., GENDREAU M., LAPORTE G., POTVIN J.Y., SEMET F., "A guide to vehicle routing heuristics", *Jour. Op. Res. Soc.*, 53-5, 512-522, 2002.

[COR 03a] CORDEAU J.-F., LAPORTE G., "A tabu search heuristics for the static multi-vehicle dial a ride problem", *Transportation Research B*, 37, 579-594, 2003.

[COR 03b] CORDEAU J.-F., LAPORTE G.,"The dial a ride problem: variants, models and algorithms", *40R1-2*, 89-101, 2003.

[COR 07] CORDEAU J.-F., LAPORTE G., "The dial-a-ride problem: models and algorithms", *Annals of Operations Research*, 153, 29-46, 2007.

[DUM 91] DUMAS Y., DESROSIERS J., SOUMIS F.,"The pickup and delivery problem with time windows", *European Journal of Operational Research*, 54(1), 7-22, Elsevier, September 1991.

[FEK 10] FEKI M.F., Optimisation distribuée pour la recherche des itinéraires multi-opérateurs dans un réseau de transport co-modal, PhD Thesis, Ecole Centrale de Lille, 2010.

[FLE 09] FLEURY E., *Réseaux de Capteur*, Dunod, 2009.

[GIL 02] GILLEN D., HAYNES M., Public and private benefits in intelligent transportation systems/commercial vehicle operations: electronic clearance and supply chain management, California Partners for Advanced Transit and Highways (PATH), Research Reports: Paper UCB-ITS-PRR-2002-18, 2002.

[HEU 08] HEUVEL A.P.R., AKKER J.M. and KOOTEN NIEKERK M.E., Integrating timetabling and vehicle scheduling in public bus transportation, Technical Report, Department of Information and Computing Sciences, Utrecht University, The Netherlands, February 2008.

[HIG 00] HIGGINS L., LAUGHLIN J.B., TURNBULL K., "Automatic vehicle location and advanced paratransit scheduling at Houston METROLift", *Proc Transport. 2000 Research Board Conference*, 2000.

[HIZ 08] HIZEM M.M. Recherche de chemins dans un graphe à pondération dynamique: application à l'optimisation d'itinéraires dans les réseaux routiers, Thesis, Ecole Centrale de Lille, 2008.

[HU 05] HU J., SHI X. ,SONG J., XU Y., "Optimal design for urban mass transit network based on evolutionary algorithm", *Lecture Notes in Computer Science 3611*, 1089-1100, 2005.

[IGO] IGO, http ://www.igocars.org/how.

[IOR 07] IORI M., SALAZAR GONZÁLEZ J.J., VIGO D.,"An exact approach for the vehicle routing problem with two-dimensional loading constraints", *Transportation Science*, 41(2), C.S. Journals, 253-264, May 2007.

[JAG 03] JÄGER W., KREBS H.-J. (eds), *Mathematics Key Technology for the Future – Joint Projects between Universities and Industry*, Springer, 2003.

[JES 08] JESPERSEN-GROTH J., POTTHOFF D., CLAUSEN J., HUISMAN D., KROON L., MAROTI G., and NIELSEN M.N., "Disruption management in passenger railway transportation", *Computers & Operations Research'*, 2008.

[KAM 07] KAMOUN M.A., Conception d'un système d'information pour l'aide au déplacement multimodal: Une approche multi-agents pour la recherche et la composition des itinéraires en ligne, PhD Thesis, Ecole Centrale de Lille, 2007.

[KON 06] KONINGS R., PRIEMUS H., NIJKAMP P., (eds.), *The Future of Automated Freight Transport – Concepts, Design and Implementation*, Edward Elgar Publishing, 2006.

[KOH 07] KOHL N., LARSON A., LARSEN J., ROSS A., and TIOURINE S., "Airline disruption management – Perspectives, experiences and outlook", *Journal of Air Transport Management*, 13(3), 149-162, 2007.

[LEC 06] LECCHINI A., GLOVER W., LYGEROS J., MACIEJOWSKI J., "Monte Carlo optimisation for conflict resolution in air traffic control", *IEEE Transactions on Intelligent Transportation Systems*, 7(4), 470-482, 2006.

[LIL] LILAS AUTOPARTAGE, http ://www.lilas-autopartage.com/savoir.html.

[LIS] LISELEC, http://www.energie-cites.org/db/la_rochelle_569_fr.pdf.

[MCB 83] MCBRIDE M.E., "Spatial competition and vertical integration: Cement and concrete revisited", *The American Economic Review*, 73(5), 1011-1022, 1983.

[MOB] MOBILITY, http ://www.mobility.ch/pages/index.cfm?srv =cms&pg =&dom=6&prub=526&rub=527.

[POC 06] POCHET Y., WOLSEY L., *Production Planning by Mixed Integer Programming*, Springer, 2006.

[PSA 95] PSAFARATIS H.N., "Dynamic vehicle routing: status and prospects", *Annals of Operations Research 61*, 143-164, 1995.

[SAV 95] SAVELSBERGH M.W.P., SOL M., "The general pickup and delivery problem", *Transportation Science*, 29, Cite Seerx, 17-29,(1995).

[SCH 78] SCHULER R.E., HOLAHAN W.L., "Competition vs. vertical integration of transportation and production in a spatial economy", *Papers in Regional Science*, 41(1), 209-225, 1978.

[STO 00] STONE J.R., AHMED T., NAVALENKO A.M. "Internet based decision support for advanced public transportation systems", *Proc Transport, Research Board Conf.*, 2000.

[TAM 91] TAMIR A., "On the core of network synthesis games", *Math Programming 50*, 123-135, 1991.

[TER] TERESE, CERTU-Lyon, Internal Communications

[VAN 04] VAN DAM K.H., OTTJES J.A., LODEWIJKS G., LUKSZO Z.V., WAGENAAR R.W., "Intelligent infrastructures: distributed intelligence in transport system control - an illustrative example", *IEEE International Conference on Systems, Man and Cybernetics*, 5, 4650 – 4654,2004.

[WAR 52] WARDROP J., "Some theoretical aspects of road traffic research", *Proc. of the Institute of Civil Engineering*, II, 1, 325-378, 1952.

[WCC] THE WORLD CARSHARE CONSORTIUM, http://www.cities.worldcarshare.com.

[WU 09] WU J., TAN Y.,"A particle swarm optimization algorithm for grain logistics vehicle routing problem", *ISECS International Colloquium on Computing*, *Communication, Control, and Management*, 364-367, 2009.

[XIA 08] XIANG Z., CHU C., CHEN H.,"The study of a dynamic dial-a-ride problem under timedependent and stochastic environments", *European Journal of Operational Research*, 185(2), 534-551, Elsevier, 2008.

[ZIP] ZIPCAR, http ://www.zipcar.com/about.