A Proposition of Data Organization and Exchanges to Collaborate in an Autonomous Agent Context

Laurent Lucien*, Christophe Lang†, Nicolas Marilleau‡, Laurent Philippe†

Abstract

We live in a world increasingly digital where intelligent and communicating objects evolve and interact. These objects have their own missions and goals. However, in a more and more complex environment, mechanisms of communication and information exchange protocols need to be more efficient, fast and smart.

The collaborative approach seems to be the most suitable to facilitate interaction between intelligent entities while preserving their resources by sharing relevant information that will enable them to reach more quickly their goals. For example, collaborative behaviours and informations exchanges could improve the movement of vehicles in an urban center and avoid traffic jams and more globally for all kind of autonomous agent.

This article highlights what is a collaboration in the context of communicating autonomous entities. Then we present our architecture that is called HACCA (Hybrid Architecture for Collaborative Communicating Agent). We show the different levels implied in HACCA and we put the stress on the multigraph structure we have created to build and share contextual knowledge. The theory is applied to an example on communication between communicating vehicles.

*PSA Group - Digital, Data & Connectivity Engineering, laurentolivier.lucien@gmail.com

†Femto-ST Institute, Univ. Bourgogne Franche-Comté, France

‡UMMISCO, IRD/UPMC, Bondy, France

1 Introduction

The digital world in which we live changes from day to day how we act, interact and develop intelligent systems allowing us to improve our lives. All these objects once designed to be used individually are now interconnected to facilitate their integration into the real world by optimizing their resources, access to information and data exchange.

An industrial and cultural revolution is underway with the development of vehicles more intelligent and communicative. Main objectives are a better safety of passengers and facilities of urban travel. For example, considering smart vehicles, platooning vehicles intend to increase security of drivers [3].

In the case of communicating vehicles, there are several possibilities of interaction: vehicle-to-infrastructure or vehicle-to-vehicle. In the first case, data is sent to the infrastructure and specific information extracted by the infrastructure from gathered data is received. But what does it happen if the infrastructure is down or unavailable (in a tunel for instance)? This interaction scheme does not however match all vehicle needs and vehicle-to-vehicle interactions are better to improve security and services to drivers for instance by exchanging information more quickly.

According to some studies [7, 10, 17, 18], there is no real autonomy if the vehicle can not fully cooperate or collaborate with any other entities. These same studies attempt to describe how these intelligent and autonomous vehicles will change our lives through three main goals: guarantee the road safety, improve the quality of life, permit the accessibility for all. So collaboration is the key of the success

of such applications. Thanks to exchanges and new ways to retrieve data, new kinds of distributed intelligence could be investigated and developed. Nevertheless, determining the collaboration is one of the major lock to design such applications. Indeed, smart objects networks are complex systems as the dynamic results from a huge amount of exchanges.

In the Multi-Agent Systems literature, several works tackle the collaboration domain [8, 12, 19, 21]. Due to the variety of mobile smart objects, a large variety of collaboration scheme exists but exchanges between connected objects are often specific to the application domain. We can note a lack of domain-expert oriented methods providing concepts and tools to qualify and study the collaboration in a complex system, such as a transportation system [13]. Thus, we tackle the problematic of collaboration and promote methods, concepts and tools to qualify exchanges between mobile entities (vehicles, drones, etc) evolving in a complex environment (city, forest). Introducing and describing collaboration between mobile entities should enhance their journey and their efficiency. For that, we take advantage of agent based systems for which the versatility allows to describe real systems by interactions between autonomous entities. In this paper, we present an hybrid agent architecture that can be used to model collaborative exchanges between mobile entities and its assessment as a autonomous vehicle model in a multi-agent system.

In section 2 of this article, we first propose a definition of the concept of collaboration and some questions we must answer when a collaboration process is implemented. According to this definition, we present related work and technical challenges in section 3 to implement a collaborative behaviour. In the next section, we present our collaborative agent architecture (HACCA - Hybrid Architecture for Collaborative Communicating Agent) which answers technical challenges. Then, in section 5, we explain how data is organized and how agents communicate data. At last, an example is presented using this organization in section 6.

2 Collaboration viewed as the refinement of Cooperation

In this section, a general analysis on collaboration is given as a preambule of a short overview about communication between agents. It permits to refine our point of view on collaboration and outline linked issues in the domain of multi-agent systems.

In the domain of Multi-Agent Systems, cooperation and collaboration concepts highlight interactions between agents and cognition: it needs some coordination actions and conflict resolution algorithms to achieve tasks [11]. Nevertheless there are differences between collaboration which is "*a form of interaction who is interested in how to distribute the work among several agents, whether it is centralized or distributed technics*" and cooperation that "*remains the prerogative of beings capable of having an explicit project therefore cognitive agents.*" [4] Collaboration is thus considered as cooperation refined by the development of a mutual understanding associated with a shared point of view of the task being solved by several interacting individuals [1, 20].

In the context of mobile objects like communicating vehicles, collaboration intends to achieve an individual mobility objective while performing a collective local task, by exchanging information between two or more mobiles. It is an intentional and cognitive process: it also results from the wishes of each mobile which collaborates with an effort of sharing selected information and a common vision of the goal to be reached.

Given this definition of collaboration, answering the following questions is a first way to illustrate and qualify collaboration in a complex system composed of smart objects:

- **Who collaborates?** – Two or more mobiles able to understand each other to exchange information or/and to share resources.
- **Why collaborate?** – To reach more easily and more quickly an objective or to satisfy a personal need or a desire.
- **When collaborate?** – When a blocking situation prevents the realization of the objectives or

when just an opportunity to collaborate arises quite simply for optimization.

- **How to collaborate?** – So that two entities are able to communicate effectively. A language (ontology), expressive enough to convey information needed to perform tasks together (e.g. express problems or plan objectives), is mandatory.
- **Which information must be transmitted?** – Any information likely to solve or help to solve a problem. It will depend on the level of confidentiality, characteristics of agent, etc.

3 Collaboration oriented architectures, overview and discussion

3.1 Overview

In the domain of Multi-Agent Systems, several architectures have been proposed to permit communication between agents or collaborative processes such as: (i) the Belief-Desire-Intention (BDI) architecture of Rao and Georgeff [16], (ii) Touring Machines of Ferguson [5], (iii) InteRRaP model of Müller [14], (iv) or DIMA of Zahia Guessoum [9]. These four architectures are summarized in the following section.

The Belief-Desire-Intention (BDI) architecture model [16] implements the principal aspects of Michael Bratman's theory of human practical reasoning with the notions of belief, desire and intention [2]. BDI architecture belongs to cognitive agent family. It introduces belief, desire intention and plans concepts for a "faithful" description of human behaviour. To manage communication, BDI architecture must be associated with another approaches (i.e Touring Machines, InteRRap and Dima) to be applied to communicating vehicle domain. The Touring Machines [5] or the InteRRaP [14] models present multilayered architectures with some specialized layers: (i) a reactive layer, (ii) a planning layer and (iii) a cooperative layer. For the InteRRap model, each layer uses a corresponding database which contains the required information for this layer. This point is interesting for our following proposition. The DIMA model [9] is a modular architecture where each module embeds all communication protocol, algorithms, reactive and cognitive subroutines. Depending on modeling, each module can be specialized. But the main problem is to handle communications transmitting between the different modules (assimilated as layers).

3.2 Discussion

All these previous models propose a multi-layer architecture with a layer for the world representation, a layer for basic behaviours, another one for planned behaviours and a final one generally dedicated for communication and/or collaborative process. Communications between agents are often limited to this dedicated layer (the higher one, the more "cognitive" layer which initiate communication if needed). This approach implies that agent do everything it can before asking help to another agent, somewhere in the simulation environment. But in cases where agents represent quick mobile objects, this consideration could be useless because the processing time would be too long. For example, if a vehicle detects an accident in front of it, it must inform immediatly vehicles all around to avoid another one.

So, for *modeling collaborative exchanges between mobile entities*, an hybrid agent architecture is required to answer quickly to environmental stimuli (reactive part). This architecture must also include some storage capacity to record experiments and a management of objectives and priorities (deliberative or cognitive part). It must be associated with a definition of its interactions with the model: the way to record information (world representation, goals, interactions between agents) and the communication protocol to use with its dependencies that depend on the multiagent development platform.

The design of this architecture also raises the following question: *how to organize data in an agent structure?* The answer however depends on the type of agents we want to create and the structure must thus be adapted to the context. We can use different possible types of representation (knowledge and facts): A simple text files with lists, an XML file for knowledge or facts, graphs (oriented or not) or

a specific structure. A good data structure allows an easy reading of the rules and therefore optimizing the determination and monitoring of targets. This work is even more necessary in the context of information exchanges (with reciprocal updates of facts and knowledge databases of communicating entities). The chosen structure of data will have to integrate all these constraints.

Concerning the *world representation* for agent, we can distinguish different possible methods: Simple variables for different states of the agent, a graph (oriented or not) for mental representation, a matrix for environment representation or an *adhoc* structure. These data structures must be chosen depending on environments. For example, a matrix is relevant for the storage of rectangular environments but a graph better fits the materialization of the links between entities. We can intend a global structure to record all linked information: world model, entities encountered, objectives and so on. A multigraph structure seems to be interesting to group all information in one only structure. So we could create links between actions, knowledge, communications history, met agents, etc to have clusters of information. These clusters could be exchanged from an agent to another.

For the *communication management*, we should use ACL for structured exchanges, in multiagent systems. It exists different possibilities: KQML [6], FIPA/ACL [15], some specific language or protocol. Whether KQML or FIPA/ACL language, we must ensure that the current vocabulary is efficient to allow the exchange of information and data between agents. The choice of the tool depends on the opportunities provided (preconditions, post-conditions, etc) and on whether the intentions are taken into account or not. We must not forget that we are in a dynamic environment where the communication time may be reduced because of the speed of travel agents. So the communication protocol must be adapted to take account of these constraints.

4 Hybrid Architecture for Collaborative Communicating Agent

In order to model a collaborative multi-agent system, we propose in this section an hybrid agent architecture based on three layers (see Fig. 1). This achitecture addresses the issues presented in the previous section. The proposition is a layered architecture where each layer is dedicated to a specific level of cognition. These layers are projected on a library of behaviour rules and the agent knowledge hierarchical database. Low level data are collected through interfaces and refined to be delivered to the analysis algorithms and data storages. The architecture also includes a communication module that implements direct inter-agent communications for a matter of flexibility and efficiency.

4.1 Layers and Behaviours

The architecture lower level is actually its interface to the environment. It is composed of two modules: the *percepts module* and the *action module*. The *percepts module* gathers information from the environment and dispatches it to both behaviour and data recording parts of the architecture. In case of a vehicle, the perception module gathers data from all the electronic sensors. The *action module* transmits orders to the various effectors of the agent. For a vehicle, this includes the management of electronic boxes that are mandatory for the link with car bodies. This lower level of the architecture also includes a raw data recording interface that sends data to the agent knowledge hierarchical database. This interface does not include an intelligent program. Data are not filtered nor sorted at this level. The purpose of this module is to facilitate the registration of raw data without the need for a specific program described in an upper layer. We explain below how data is organized and how it is used.

Three behaviours levels operate the collaboration according to behaviour cognition graduation. The first layer (*reaction*) manages all reactive behaviours. This is the operational module where all basics re-

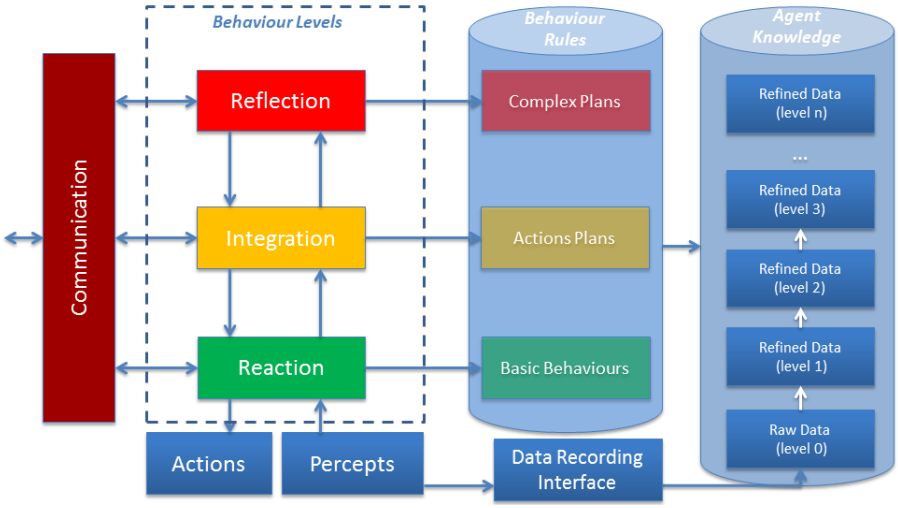

Figure 1: HACCA: Hybrid Architecture for Collaborative Communicating Agent

actions are treated. The second layer (*integration*) brings some more refined behaviours with a deepening of the decision process. This is the tactical module where actions plans with special constraints are implemented. The third layer (*reflection*) allows to integrate complex behaviours based on the previous layers decision process. This is the strategic module with some long-term views and where complex missions to be accomplished are implemented. The different layers are requested by message or by program (this may vary depending on the implementation, we will discuss about this in the next section). For example, the *reaction layer* depends on the data gathered by the perception module. Depending on the programmed basic behaviours, the *integration layer* can be called to execute subroutines helping the realization of the complete task. Similarly, the *reflection layer* can be called by the program of the integration layer if needed. All communications between layers can be associated with treatment priorities. This operation seems to be most appropriate to manage the behaviour of an intelligent vehicle that must sometimes quickly respond to environmental events while also being able to execute more complex tasks requiring to go through the basic knowledge and to establish action plans. The three layers use the *behaviour rules* library and the *agent knowledge* database.

4.2 Rules and Knowledge Representation

The *behaviour rules* library is a hierarchical object. The atomic bricks are the *basic behaviours* of agents, their reactions to the environment perception. Each *basic behaviour* has its own preconditions, constraints and postconditions. Then we find the *actions plans* which are composed by *basic behaviours*. Finally there are *complex plans* composed by *actions plans*. For all actions or plans, we can also find preconditions and constraints. All these *basic behaviours*, *actions plans* and *complex plans* are linked together. The organisation of these links are like a tree of actions. Each *basic behaviour*, *actions plan* or *complex plan* is called by a specific layer that is a kind of entry point in the action tree. This leads to a basic work can be conditionned by the *reaction layer* and possibly continue the whole plan by triggering the top layer. This is very useful when modeling highly mobile agents like communicating vehicles crossing at high speed on a road. The detection process (driving by percepts) executes the basic action but if the information to be exchanged is relevant to the other

vehicle, an additional request using a structured action plan can be called. This structured action or plan will be a part of the global tree of actions.

The main objective of the *agent knowledge* database is to correctly structure data to manage them quickly and communicate them to the outside. For example, sensors collect raw information that will be stored in the knowledge base. Then this raw information is analyzed and treated gradually by actions attached to the different layers. We can attach other information to complete and refine basic information. As to treatment, we obtain an information tree with basic information on the root and all information attached to the branches. These branches can be achieved by the various layers according to the requirements of the treatment.

Finally, we use a multigraph structure (which is described in the section 5) to ensure links between data, behaviours, agents, interactions and so on. This structure allows to manage all data relevant to a given action. Therefore there is a link between the requested action, the level of information we want to communicate (basic information or information more or less refined), the contacted agent, the results of the exchange. All these elements form a coherent whole. The nodes of the multigraph become entry points to a set of structured data. This data structure is useful to manage a huge amount of information. It also allows not to waste time during exchanges focusing on the right level of information (by choosing the right node of the tree). However all data remains accessible if the situation requires it. Each changing context adds new links and new nodes to the original structure.

4.3 Communications Management

For us, we can define some collaborative actions and communications at different level. The *communication module* is the most important part of this architecture. To ensure a high level of responsiveness, this module is transverse. The *communication module* can be called by any layer, selects the right protocol depending on the geographical distance and the assigned task (diffusion, partial collaboration, full collaboration) and maintains links with the listener as long as the mission requires, so lets move from one protocol to another as needed. For example, the agent detects an anomaly in its environment. A *basic behaviour* commands to inform agents all around but also to seek and define the nature of the anomaly. Then the agent uses the *communication module* for dissemination of critical information. Meanwhile, the *basic behaviour* triggers an *action plan* in the upper layer to understand what is the anomaly. Agents all around will receive the message that a critical element has been detected. Some agents (according to their objectives) will return a message to the first agent for more information. The message is received by the *communication module* but the message is routed to a top layer, the one that deals precisely the process of analysis of the anomaly. This block allows monitoring of the communication process between agents. Initially, there is a generic communication with a diffusion process which will turn into specific communication between two agents. As the *agent knowledge* base which is a multigraph, the communication process will be part of the same logic linking information and communicating agents. This module allows for easier building of a structured exchange based on hierarchical data.

5 Multigraph for building and sharing Contextual Knowledge

In the previous section we introduced quickly the agent architecture. Here we describe the organization of the agent knowledge.

In the context of mobile agents operating in an environment, there are mainly two types of observed or recorded items: static or stationary elements (in the environment, for example a road network), dynamic or movable elements (such as vehicles moving on this road network).

All agents interact continuously throughout the simulation. As we are in a context of autonomous agents possessing a level of reflection, it is necessary to record the events they face. According to the architecture that we described in the previous section, the agent has several levels of interpretation (reaction

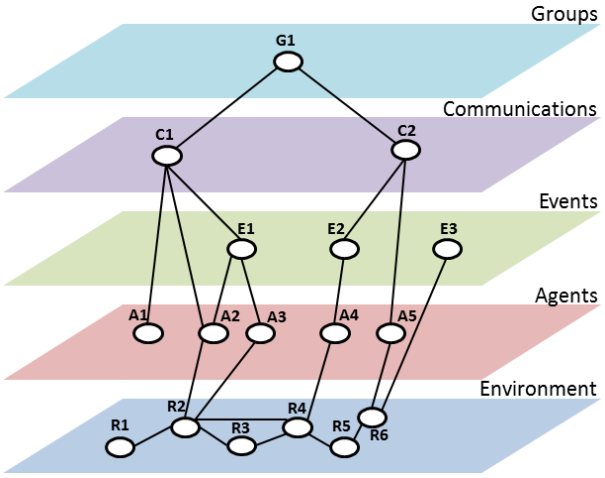

Figure 2: Representation of the multigraph structure (with 5 layers of objects and different kinds of nodes)

or operational, integration or tactical, reflection or strategic). Information collected in the environment must be stored in the agent knowledge in order to be used later by different layers with their associated behaviours. This means that during the simulation, the recorded information will gradually be refined, modified or deleted depending on programmed behaviours.

In addition, if the current agent finds itself in a situation where it does not have any solution or meets a particular situation of immediate danger or other information to communicate quickly, it must be able to communicate it to other agents. As we are in a collaborative context and in accordance with the definition that we have given in section 2, each information must be contextualized (if the need is proven). This implies that it is important to convey the sent information as well as sufficient information for another agent to understand properly and interpret correctly the received information.

In order to prioritize information properly, we propose a multigraph data organization (or multilayer) where each layer represents an object type (see Fig. 2). The first layer of the graph represents the environment or global static objects known or met by agent. The second layer is dedicated to dynamic objects (for example agents) that the current agent can observe and deal with them. The third layer represents events that the agent perceives or infers from observations. The fourth layer is used to track interactions with other agents recording the ongoing communications. The fifth layer symbolizes the concept of group and collaborative actions. All objects shown are nodes of different graph layers. At each stage of observation, the agent identifies the environmental elements (static or dynamic) and saves them in the multigraph then establishes links between elements. The various actions or behaviours gradually alter or enrich the multigraph.

Each object has common properties: a unique identifier, a designation (or nature of item, for example a road or a vehicle), a location, an observation date and a status (element state at observation time). However, depending on the nature of each object, additional properties can be recorded. For example, for a vehicle object, we find the path or travel time between point A and point B or the average speed on the daily trip. Another example for a communication object, there is the geographical area of emission (local broadcast, broadcast over an entire area, etc.), the priority of message or the time out between two messages.

For example, in an urban context and for a vehicle agent, the layer Environment is the road network (see Fig. 2., Nodes R1-R6). Each node of the graph represents a road or an intersection (it is not necessarily required to model the intersection if it does not include specific items used by agents). The layer Agents records other vehicles observed at each step of the simulation (nodes A1-A5). Each node vehicle will be linked to the road node it occupies (for example, link between A4 and R4). If several vehicles are stopped on the road of the agent, each vehicle is symbolized by a specific node (eg, nodes A2 and A3). Then a further process (depending on possible behaviours) can identify or infer a traffic jam (node E1). This traffic jam may be caused by a damaged vehicle which will also be identified (vehicle: node A4 accident: node E2). Traffic jam and accident are two examples of events recorded on the layer Events of the multigraph structure. At this level, links connect events nodes to vehicles nodes, themselves con-

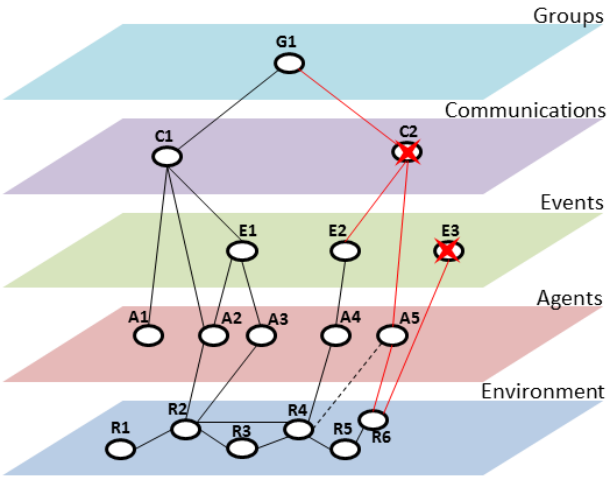

Figure 3: Evolution of the multigraph structure (example with the end of communication C2 and event E3 treated)

nected to roads nodes. As part of the accident (node E2), the vehicle that detected the event will inform other vehicles of its scope of communication (to be defined depending on the model), eg vehicle node A5. Each communication is a new node on the layer Communications of the multigraph (here node C2) and links are created between the communication node, the event node and vehicles nodes concerned by this communication (links between node C2, node E2, and node A5). Another example, node C1 represents an ongoing communication between current agent, node A1 and node A2 about the event E1. If the behaviour requires a more successful partnership, not a simple communication or intervention of several vehicles (eg, creation of a group), a group node is created on the layer Groups of the multigraph. Links are then established between the node group and communication nodes to symbolize the group and exchanges between different members (node G1).

During the process and successive interactions, the multigraph structure evolves. Some links are created, others are moved or deleted. Similarly, nodes can be updated or deleted if the context has changed (see Fig. 3).

For example, the communication between current agent and agent A5 is completed. The communication node C2 is deleted. Obviously, the link connecting node C2 to node A5 is also deleted even the link to the event that caused the communication (between node C2 and node E2). Another case, the agent A5 moves on another road section (R6 to R4). The link that connected node R6 to node A5 is also removed. Another example, the event E3 symbolizing a blocked road is no longer valid. The node E3 is removed even links with road node R6.

Be aware that it is the symbolic representation of world as the agent perceives it with the information it has collected or has exchanged. This is not a real-time view of the situation. Some data may be incorrect if the observation date is too old hence the importance of ownership of observation date stored in each data node. Some behaviours can incorporate cleaning instructions of the structure. For example, if the difference between the current system date and the observation date exceeds a certain duration. In addition, these dates will necessarily be evaluated by agents receiving the information. Depending on the date, data can be simply ignored.

To summarize, the multigraph structure can be modified by environmental perception, exchanges of information, execution of behaviours, periodic or event-based cleaning behaviours retained by the agent to accomplish its objectives.

After the presentation of data structure, we describe how these data are exchanged during interactions between agents. We must distinguish several cases.

If the agent must communicate quickly crucial information, for example, warn following vehicles that an accident occurred just in front and an emergency braking is required, we can not send a lot of information frames on multiple communication channels. The message must be short and clear. For the agent watching the scene of accident, the behaviour related to the perception process generates the creation of an event node recording of the accident and connects it to the node representing the current road of agent. The generation of this kind of event triggers a diffusion procedure to inform vehicles in the geographical area. The message only contains main information (accident) and order of propagation (see Fig. 4).

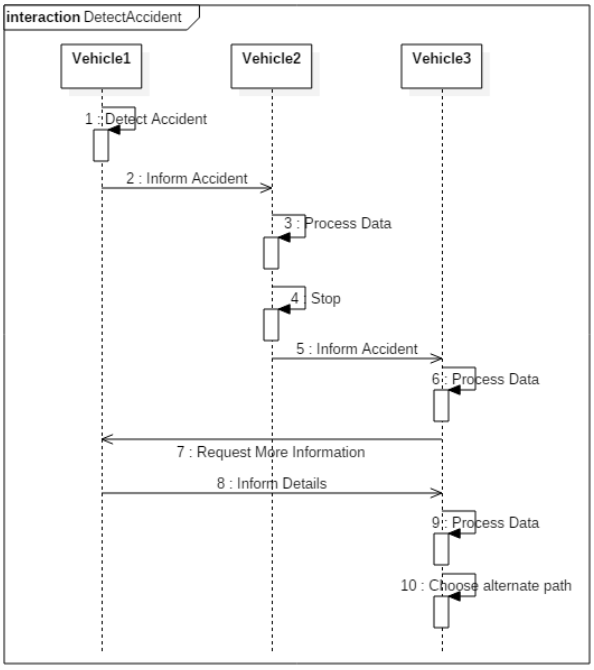

Figure 4: Sequence diagram for a detection of accident and communications for exchanges

Vehicles agents around the sending agent vehicle receive information that also triggers an immediate return in appropriate behaviour that is processed by the operational layer of the receiving agent (communication between Vehicle1 and Vehicle2). Information is also integrated into the multigraph of each agent but unrelated to a road node or another agent node (involved in the accident). This is basic information to enable immediate reaction (here a stop of Vehicle2 and propagation of the first alert message). Secondly, based on programmed behaviour, vehicle agent can ask in return information on the context of the accident. The first vehicle agent that issued the alert can therefore send a complete message by extracting a portion of the multigraph structure (communication between Vehicle3 and Vehicle1). As mentioned in the previous paragraphs, the event node is connected to a node agent that is connected to a road node. This helps to contextualize event precisely with these characteristics. This extract of the structure can change the behaviour of the recipient agent, for example, trigger an avoidance procedure if it were actually use the road where the accident occurred but it has time to change path.

To formalize a message frame, we use FIPA [15]. In any case, we find a unique identifier for conversation (which will be used for all correspondence between agents and serve as a reference), a sender, a receiver, a subject and a content. In the previous example, we have two types of messages: inform and request. The message content is an extraction of the multigraph structure. We describe mecanisms in the next section with an example in an urban context.

6 Example in an Urban Context using HACCA

Figure 5 presents a concrete case using HACCA (see section 4). Several vehicles (vehicle1 to vehicle8) evolve on a road network (road1 to road6). Each vehicle follows a particular path to get from point A to point B. Here all vehicles except vehicle8 follow road1, road3 and road6. Vehicle8 takes road4 then will take road2 and road5 before taking road6. The

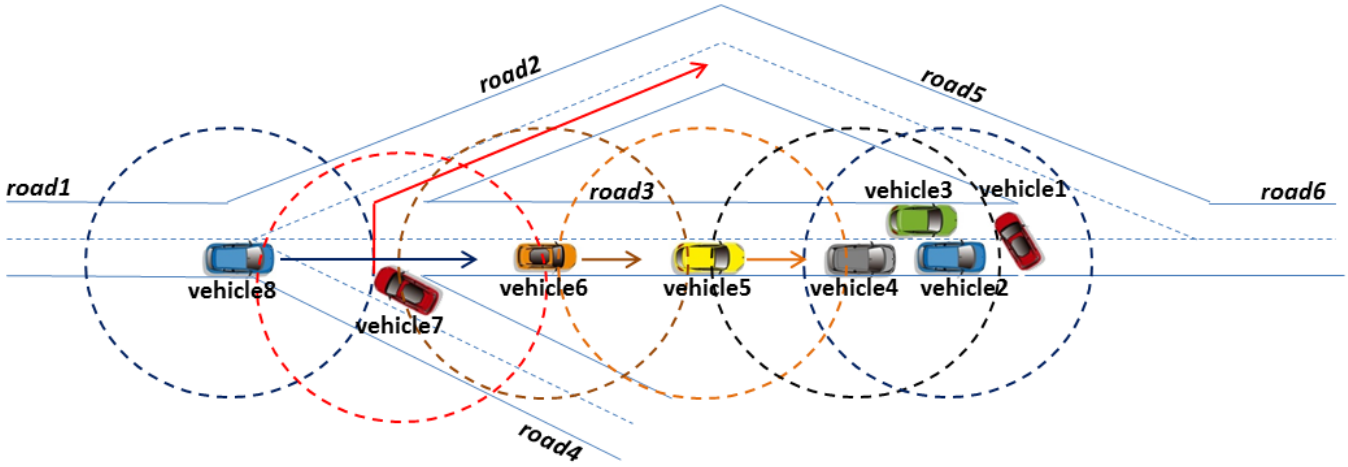

Figure 5: Example with several vehicles (communication areas and paths are represented)

circles symbolize the boundary of the field of communication.

To explain mechanisms of organization and data exchange, we will focus on vehicle2, vehicle4, vehicle7 and vehicle8. Vehicle2 observes the accident and stops. Vehicle4 does not observe the accident but stopped cars in front of it. Vehicle7 is *a priori* not affected by the accident. Vehicle8 is not yet on the accident area and can potentially avoid it.

What is happening to vehicle2? Vehicle2 observes its environment through the perception module (composed of sensors). It observes the scene all around and finds two stopped vehicles (vehicle1 and vehicle3) and also stopped the vehicle following (vehicle4). If we take the detailed structure in the section 5, vehicle2 already has in the multigraph structure, a definition of road network in the layer Environment. The Data Recording Interface module records observations in the layer Agents of multigraph. This triggers execution of a behaviour of the layer Reaction which will interpret situation. Vehicle2 concludes that vehicle1 had an accident and road3 is blocked. The multigraph structure will be enriched by two events in the layer Events. Both events trigger transmission of two messages. The first will notify nearby vehicles there is an accident without more information and they must stop absolutely. The second is a more general message to signal that road3 is blocked. The communication module, considering the importance of messages, will propagate information of accident with the highest priority and demand spread to the peripheral area to prevent further accidents. The message will be short and imperative. Information from the blocked road will be sent with a lower priority but with a much wider distribution to allow following vehicles to alter their path. Both communications are recorded in the multigraph by linking events, vehicles and concerned road.

What is happening to vehicle4? Vehicle4 observes its environment through the perception module. It notices two stopped vehicles (vehicle2 and vehicle3) and a moving vehicle following it (vehicle5) but it does not see the accident. The Data Recording Interface module records observations in the layer Agents of multigraph. This triggers a behaviour in the layer Reaction which interprets the situation. Vehicle4 concludes that there is a traffic jam made of vehicle2 and vehicle3, road3 is blocked. This triggers an update of the multigraph structure by creating two events in the layer Events. Both events trigger transmission of two messages. The first will notify nearby vehicles there is a traffic jam and must reduce their speed. The second message informs that road3 is blocked. Both messages will be broadcast on the

geographical area with a request to spread without warning message or a very high priority. Moreover, in the case of traffic jam, the message will contain the event description with a list of stopped vehicles and the road concerned. Both communications are recorded in the multigraph by linking events, vehicles and the road concerned. The content will be an extract of multigraph with traffic jam node as entry point. Going up in the graph structure, we get all information. Vehicles receiving the message can decide reaction based behaviours built into agent. In addition, via the communication module, vehicle4 receives messages sent by vehicle2 in particular information about an accident with a high priority and an order of propagation. This will trigger a behaviour in layer Reaction and vehicle4 will stop immediately. At the same time, it spreads information on the accident. At this stage, two messages were sent by vehicle2 (accident and blocked road), three by vehicle4 (traffic jam, accident and blocked road). Vehicle4 already known the road blocked so it strengthens the knowledge of agent.

What is happening to vehicle7? Vehicle7 observes its environment through the perception module. It observes two moving vehicles (vehicle6 and vehicle8) and records information via the Data Recording Interface module. *A priori* vehicle7 is not concerned by the accident, nor by the blocked road because it must take road2. However, via the communication module, it will receive broadcast messages about traffic jam and blocked road that were issued and then redistributed by vehicle5 and vehicle6. The message about accident is no longer broadcast because vehicle7 is already quite far from it. After updating its multigraph structure and the initiation of appropriate behaviour, vehicle7 concludes that it is not concerned at all. But as messages contain a request to spread over a wide geographical area, it rebroadcasts around it the two received messages.

What is happening to vehicle8? Vehicle8 observes its environment through the perception module. It sees only vehicle7 moving. It records observations in the multigraph structure via the Data Recording Interface module. It also receives through the Communication module, two incoming messages informing of a traffic jam and a blocked road. The update of multigraph internal structure triggers appropriate and more elaborate behaviour of the layer Integration. As vehicle8 is affected by road3, it seeks a potential new path to avoid area. According to its knowledge, road2 can be taken. To acquire a certainty, it sends a message around to ask for information. So far, all sent messages used the FIPA inform type. Here, the message type is request. This creates a new communication node in layer 4 of the multigraph. As vehicle7 is in the reception area, it processes this message and looks into its internal structure if road2 is free. Then a dialog is engaged between the two vehicles to exchange information requested by updating the multigraphs structures of each.

In the end, after the accident with vehicle1, vehicle2 and vehicle3 were able to stop just in time and prevent following vehicles including vehicule4 to prevent further accidents. Then vehicle5 and vehicle6, on the same road have been informed of the creation of a traffic jam so they can gradually slow down and adjust their speed to the situation. Vehicle7 was not involved but has an information dissemination work (communication relay). Finally vehicle8 has read information on the geographical area and has established an alternative path with the help of vehicle7 information.

7 Conclusion

In this paper, we defined the collaboration between mobile agents. According to this definition and the urban context, we proposed HACCA, a multiagent hybrid architecture with three dedicated layers (reaction, integration and reflection). These layers are projected on a library of behaviours rules adapted to all situations (emergency, consolidation, distribution, etc) and an agent knowledge. Then we described the way data is organized in a multigraph structure and how to exchange information between agents. At last, an example with communicating vehicles is explained using this new structure and architecture.

We will continue the development of the collaborative agent architecture by implementing the full multigraph concept (for actions and behaviours) and by improving the information dissemination process

in connection with the communication block. We will establish more elaborate action plans in connection with road traffic to show the interest of structured communication between the communicating vehicles and smart infrastructure. BDI mechanisms will be added to organize the internal decision processes and allow the creation of joint plans between agents.

Aknowledgements

This work is sponsored by PSA group as part of the executive PhD program.

References

[1] M.-F. Blanquet. Web collaboratif, web coopératif, web 2.0.: quelles interrogations pour l'enseignant documentaliste. *Formation des personnes ressources en documentation*, 2009.

[2] M. Bratman. *What is Intention?* Number n 27 ;n 69 in Report (Center for the Study of Language and Information (U.S.))). Stanford University, 1987.

[3] Jean Michel Contet, Franck Gechter, Pablo Gruer, and Abder Koukam. Multiagent system model for vehicle platooning with merge and split capabilities. In *Third International Conference on Autonomous Robots and Agents*, pages 41–46, 2006.

[4] Jacques Ferber. *Les systèmes multi-agents : vers une intelligence collective*. Informatique, Intelligence Artificielle. InterÉditions, 1995.

[5] Innes A. Ferguson. Touring Machines: Autonomous Agents with Attitudes. *Computer*, 25(5):51–55, May 1992.

[6] Tim Finin, Richard Fritzson, Don McKay, and Robin McEntire. KQML as an agent communication language. In *Proceedings of the third international conference on Information and knowledge management*, pages 456–463. ACM, 1994.

[7] Bernard Flury-Hérard and Hervé De Tréglodé. Les véhicules communicants nécessitent-ils de nouvelles réglementations, 2015. 2015.

[8] Barbara J Grosz. Collaborative Systems (AAAI-94 Presidential Address). *AI magazine*, 17(2):67, 1996.

[9] Zahia Guessoum. *Un environnement opérationnel de conception et de réalisation de systèmes multi-agents*. PhD thesis, Paris 6, Grenoble, 1996. Th. : informatique.

[10] Ratan Hudda, Clint Kelly, Garrett Long, Jun Luo, Atul Pandit, Dave Phillips, Lubab Sheet, and Ikhlaq Sidhu. Self driving cars. *College of Engineering University of California, Berkeley, Berkeley: College of Engineering University of California, 2013*, 2013.

[11] N. R. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. *Artificial Intelligence*, 75(2):195 – 240, 1995.

[12] Steven KC Lo. A collaborative multi-agent message transmission mechanism in intelligent transportation system–a smart freeway example. *Information Sciences*, 184(1):246–265, 2012.

[13] Julien Monteil, Romain Billot, and Nour Eddin EL FAOUZI. Véhicules coopératifs pour une gestion dynamique du trafic: approche théorique et simulation. *Recherche Transports Sécurité*, (29):pp–47, 2013.

[14] Jörg P. Müller and Markus Pischel. The Agent Architecture InteRRaP: Concept and Application. Technical report, 1993.

[15] Stefan Poslad, Phil Buckle, and Rob Hadingham. The FIPA-OS agent platform: Open source for open standards. In *Proceedings of the 5th International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents*, volume 355, page 368, 2000.

[16] Anand S. Rao and Michael P. Georgeff. BDI Agents: From Theory to Practice. In *IN*

PROCEEDINGS OF THE FIRST INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS (ICMAS-95, pages 312–319, 1995.

[17] Gary Silberg, M Manassa, K Everhart, D Subramanian, M Corley, H Fraser, and V Sinha. Self-driving cars: Are we ready. *KPMG. October 2013*, 2013.

[18] Gary Silberg, Richard Wallace, G Matuszak, J Plessers, C Brower, and D Subramanian. Self-driving cars: The next revolution. *White paper, KPMG LLP & Center of Automotive Research, 2012*, 2012.

[19] Bastin Tony, Roy Savarimuthu, and Maryam Purvis. A collaborative multi-agent based workflow system. In *Knowledge-Based Intelligent Information and Engineering Systems*, pages 1187–1193. Springer, 2004.

[20] Gerhard Weiss. *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT press, 1999.

[21] Minjie Zhang, Quan Bai, Fenghui Ren, and John Fulcher. Collaborative Agents for Complex Problems Solving. In ChristineL. Mumford and LakhmiC. Jain, editors, *Computational Intelligence*, volume 1 of *Intelligent Systems Reference Library*, pages 361–399. Springer Berlin Heidelberg, 2009.