Cloud-Assisted Distributed Control System Architecture for Platooning

Umberto Montanaro
and Saber Fallah
Connected and Autonomous Vehicle Lab
Department of Mechanical Engineering
University of Surrey
Guildford, UK, GU2 7XH
Email: s.fallah@surrey.ac.uk

Mehrdad Dianati
WMG, International Manufacturing Centre
University of Warwick
Coventry, UK, CV4 7AL

David Oxtoby
Tom Mizutani
and Alexandros Mouzakitis
Jaguar Land Rover Limited
Coventry, UK, CV3 4LF

***Abstract*—This paper presents a functional architecture for controlling and managing platoons of vehicles assisted by a cloud computing platform for passenger vehicles in mixed traffic scenarios. The architecture modules are distributed in the three separate logical layers suggested by the on-going CARMA (Cloud-Assisted Real-time Methods for Autonomy) research project which aims to develop and test cooperative automated driving technologies and distributed control systems. The architecture also exploits the potential of the CARMA platform both in terms of computing and data sharing.**

I. Introduction

A vehicle platoon is a group of two or more consecutive Connected Autonomous Vehicles (CAVs), traveling along a highway in the same lane and at the same velocity with a short inter-vehicular distance. Platooning of vehicles can offer benefits such as safety, efficient road usage, efficient fuel consumption and passenger comfort [1]. The achievable benefits promised by this cooperative driving vision have been investigated by research projects such as SARTRE [2], COMPANION [3], Energy-ITS [4], Connected & Drive [5], just to name a few. To guarantee a coordinated motion of the vehicles in the platoon, cooperative adaptive cruise control (CACC) algorithms are used to compute the acceleration of each vehicle based on on-board measurements and information gathered from the other platoon members through Vehicle-to-Vehicle (V2V) communication systems [1]. However, for the practical implementation of a vehicle platoon, the use of the sole CACC systems and V2V communication technology is not sufficient. Other platoon functionalities, such as those required to execute basic platoon manoeuvres (e.g. merging/leaving the platoon), must be integrated and properly orchestrated to guarantee the correct cooperation of the platoon members. Furthermore, off-board information such as road regulations and roadway conditions should be provided to platoons through Vehicle-to-Infrastructure (V2I) communication systems to increase the perception of the surrounding environment and improve safety [6]. To achieve the desired platoon vision, functional architectures are often used to describe platoon functionalities and to define the data-flow among the system control components. Functional architectures usually are composed of different layers which are distributed between on-board vehicle systems and a roadside infrastructure such as servers, cloud, etc. Each layer of the architecture is built up of different functional modules and is responsible for a set of activities. The earliest platoon architecture was proposed within the PATH program [7], which consists of five layers: physical, regulation, coordination, link and network layers. The first three layers reside in the vehicles, while the link and network layers are distributed to the infrastructure. The physical layer consists of control modules for the vehicle, such as steering, brakes, transmission control and local sensors. The regulation layer is used to plan and execute local manoeuvres required by the upper layer. The coordination layer plans the manoeuvres to be actuated and is supported by information gathered from the other vehicles and the infrastructure. The link and network layers are used to control the traffic flow. The link layer is devoted to control traffic over sections of the highway (up to 5 km), while the network layer controls traffic flow over the entire highway network and plans vehicle routes to maximise the capacity or minimise the average vehicle travel time. Another example of architecture for platooning was proposed in [8] where the architecture was structured in three layers, i.e. traffic control layer, vehicle management layer and vehicle control layer. The vehicle control layer is responsible for sensing the environment and controlling the vehicles whereas the vehicle management layer coordinates movements of vehicles in platoons and the traffic control layer provides instructions for optimising the traffic flow along the highway. Similarly, a three-layered architecture was suggested in [9] to develop either a centralised management or a decentralised management. The former refers to a platoon system in which the platoon leader, i.e. the

first vehicle in the platoon, takes decision and coordinates the other platoon members during platoon operations while the latter refers to platoons in which platooning decisions, such as those required for merging/leaving manoeuvres, are distributed among platoon members without the need of the platoon leader. It should be noted that all abovementioned architectures have been developed based on the assumption of automated highways in which all vehicles are assumed to be connected and autonomous. However, the hypothesis of having all road vehicles to be CAVs is not realistic for the short-term future [10]. Consequently, platoon architectures suitable for mixed traffic scenarios (i.e. traffic scenarios that in addition to CAVs include autonomous vehicles, connected vehicles and conventional vehicles) are of utmost importance. Recently, a platoon functional architecture that can be operated in mixed traffic has been proposed in [3] for goods transportation. The architecture is composed by three layers: strategic, tactical and operational layers. The strategic upper-layer assists a dispatcher at a carrier company to schedule the trip of the vehicles of the company such that goods are delivered within a desired time and with the minimum fuel consumption. To achieve these targets, the strategic layer optimises the route and speed of each connected vehicle and organise, when possible, sets of vehicles in platoons while considering real time traffic data which can be obtained without assuming that all road vehicle are CAVs. It is noted that the computation of the routes allowing organisation of vehicles in platoons was possible because the transport assignment of all vehicles (i.e. start and destination of the trip and desired time of delivery) was assumed to be known before an actual transport occurs. The tactical layer handles short term goals, such as refinement of speed profile either for fuel minimisation or for guaranteed platoon formation. Finally, the operational layer performs activities similar to those provided by the coordination and regulation layers of the architecture proposed in the PATH program [7]. In the architecture proposed in [3], the strategic layer is implemented in the infrastructure while the operational layer resides on-board the vehicle and the tactical layer is partitioned in both the off-board and the on-board systems.

In the framework of platoon architectures, the aim of this paper is to present a distributed functional control platoon architecture for passenger CAVs that has the potential to be assisted by the cloud computing framework under development within of the on-going CARMA (Cloud-Assisted Real-time Methods for Autonomy) research project [11]. The platoon architecture is designed for CAVs connected to the CARMA platform and can be used for a mixed traffic highway. Moreover, the architecture is intended to support passenger vehicles while on highways to create, form and maintain platoons. Drivers can decide at any time to initiate or end platooning with other CAVs connected to the platform, thus extending the highway scenario presented in [3], where the formation and duration of platoons in the platform can be scheduled offline and before the actual trip of vehicles. Furthermore, the architecture is designed to be distributed over the layers of the CARMA architecture to benefit from the computational power and information available within the CARMA framework. The paper is devoted to present an overall view of the entire platoon control architecture under investigation in the CARMA project together with its fundamental blocks and it is organised as follows. In Section II, an overview of the CARMA project with a description of the layers of the CARMA architecture as provided in [11] is given to clarify how the platoon architecture is distributed through the CARMA framework. In Section III, the highway scenario which includes the organisation of the CAVs connected to the CARMA platform in platoons is provided with the aim to give the reader an overall description of the functionalities to support cooperating driving in highways. Components required to implement such a scenario are provided in Section IV, where the architecture for platooning is presented and distributed through the CARMA layers. Finally, conclusion and future research lines are drawn in Section V.

II. The CARMA Project: An overview

CARMA is a research project co-funded by the UK's Engineering and Physical Sciences Research Council (EPSRC) and Jaguar Land Rover under a programme of five projects collectively called "Towards Autonomy - Smart and Connected Control" where the automation focus is BASt Level 3+. The aim of CARMA is to develop and test cooperative automated driving technology, based on a distributed control system, which is enabled by an ultra-low latency and highly reliable cloud-based infrastructure accessed through 5G. The CARMA approach is that of a distributed control system in which the executions of autonomous functions are distributed between the on-board system and a cloud-based high performance shared back-end system. The CARMA architecture is logically divided into three layers, as depicted in Figure 1, i.e. the CARMA Vehicle, the CARMA Edge and the CARMA Core [11]. The CARMA Vehicle connects various on-board sensors, infotainment equipment, on-board embedded processors, Human-Machine Interface (HMI) equipment, and actuators to apply control commands. The on-board control components operate in cooperation with the CARMA Edge while assuring fault-tolerance of the system in cases when the connection is disrupted. Furthermore, since vehicle safety is of paramount importance, on-board controllers are also responsible to assess and potentially override the remotely computed instructions (from cloud/edge) to ensure safety of

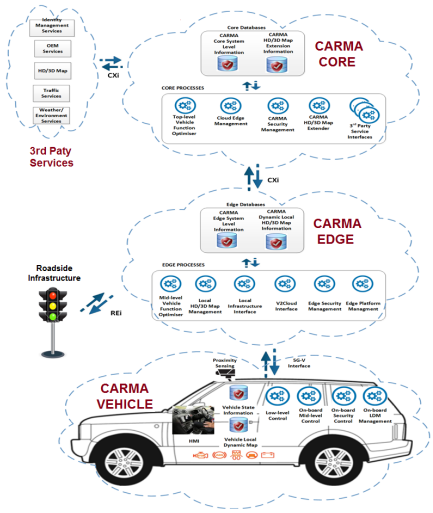

Fig. 1. CARMA 3-tier logical architecture (taken from [11]).

the vehicle. The CARMA Edge hosts off-board processes and information that require tight access (low latency) with the vehicles. This will include information collected from around the vehicle and processes that require cooperation with roadside equipment and other vehicles. This concept is borrowed from ETSI Mobile Edge Computing (MEC) framework, where MEC servers can be installed at Road Side Units or cell sites. The CARMA Core is a cloud-based back-end system, based on commercially available public cloud resources. This also provides interfaces for a variety of stakeholders that provide or use information from CARMA vehicles.

III. Highway Scenario

This section describes a CARMA vehicle's trip through a highway network from entry to exit. During the trip, the vehicle can join one or more platoons, thus it will be also described how platoons are formed and managed within the CARMA platform. For the sake of simplicity, the highway scenario is detailed in the nominal case in which there is no loss of connection or failure of on-board vehicle sensors/actuators. It is also assumed that conventional vehicle cutting in to enter in the platoon will not happen. The highway system can be represented as a set of highway segments where each is composed by the highway name, entry and exit gates [7]. Each segment is divided in sections, where each section is defined as a piece of a highway segment belonging to the coverage area of a CARMA edge. It is assumed that each highway segment is a sequence of either consecutive or overlapping sections, thus guaranteeing that the CARMA vehicle is always connected to the CARMA platform. A schematic representation of a highway segment composed of

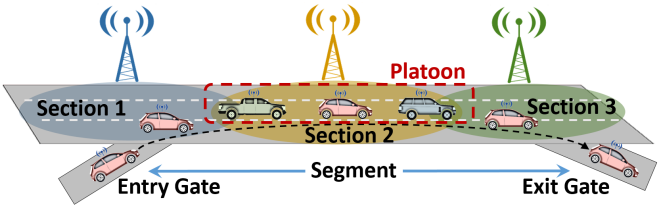

Fig. 2. Highway scenario.

three sections is given in Figure 2. Upon entering a highway gate, the CARMA vehicle announces its destination together with a set of driver's preferences for the computation of the vehicle route. In respond, the CARMA platform computes, according to some criteria (e.g. minimum path, minimum fuel consumption, etc.), the best route connecting the entry gate to an exit gate. The route is described as a sequence of highway segments and their sections. For each section the CARMA platform provides to the CARMA vehicle the speed profile that minimises the fuel consumption over the entire trip (i.e. global optimal speed). This speed is actuated through an on-board longitudinal control. A CARMA Vehicle travelling along road sections and following the suggested global optimal speed is said to be operated in ACC mode by definition. During the trip, the driver of the CARMA vehicle can decide to initiate platooning with other CARMA vehicles sharing with it a sequence of road segments. Hence, the driver sends to the platform the request either to create a new platoon or to merge to an existing one. If the driver has requested the creation of a platoon, a new platoon is created in the CARMA platform and the CARMA vehicle is set as leader of the new platoon and the state of the vehicle changes from ACC mode to Platoon Leader (PL) mode. If the driver has requested to join a platoon, the state of the vehicle changes from ACC mode to Free Agent (FA) mode by definition. In this state, the CARMA platform provides the vehicle with a set of platoons available in the system that are compatible with the speed profile of the CARMA vehicle, i.e. platoons which are reachable according to the speed profile previously computed. Furthermore, each platoon in the list is characterised by a set of features (e.g., achievable fuel reduction driving in that platoon, expected time to reach the platoon, etc.) which supports the driver in the selection of a platoon to join (such platoon is referred in the rest of paper as target platoon). It is noted that, due to the variability of the traffic, the platform continuously checks if the selected platoon is reachable, if not, it informs the driver and suggests possible alternatives. When the CARMA vehicle in FA mode is close enough to the target platoon, a pre-merging phase is initiated. In this phase, the CARMA platform supports the CARMA vehicle to perform a set of activities that must

precede the merging of the new member in the platoon, such as (i) the selection of the correct lane, (ii) selection of the position of the new member within the platoon (e.g., to keep vehicles in the target platoon sorted according to a given criteria), and (iii) the computation of suitable speed profiles to reduce the gap between the CARMA vehicle and the target platoon. It is noted that, in this phase both the velocity of the CARMA vehicle and that of the target platoon can be adapted to guarantee that their relative position is adequate before the merging starts [12], [13]. Furthermore, the platoon can be configured either by considering the destination of each vehicle in the platoon so as to maximise the time the platoon stays intact [14], or by the ability of each vehicle to actuate acceleration/deceleration so as to put vehicles with poorer capability at the end of the vehicle string [15]. The subsequent operations that the CARMA vehicle and the target platoon perform are similar to those presented in [16], i.e. merging platoon, platooning, and leaving platoon. Briefly, during the merging phase, the FA vehicle reduces further the relative distance with respect to the platoon[1] and performs a lane change in the case of a side merging after the target platoon has prepared a suitable gap for the new member. After the merging phase is completed, the state of the CARMA vehicle shifts from the state FA-mode to either PL mode (if it is the first vehicle in the platoon) or Platoon Follower (PF) mode (if the vehicle is not the first in the string). During platooning, all the vehicles in the string move at the same speed of the leader with a precise control of the inter-vehicular distance guaranteed through the use cooperative adaptive cruise control (CACC) algorithms [1]. Finally, during the leaving phase, the CARMA vehicle exits from the platoon by performing a lane change when its lead and following vehicles have created a suitable gap for executing this manoeuvre. A CARMA vehicle leaves the platoon either on the request of the driver or when its route becomes different from the platoon route. It is noted that, the CARMA platform supports all these manoeuvres by orchestrating their activation and providing ambient data (e.g. road parameters, information on possible adjacent vehicles etc.) for the correct and safe execution of manoeuvres. Moreover, during platooning, the CARMA platform optimises the speed and inter-vehicular distance of the platoon to reduce fuel consumption during the cooperative driving [17]. After a CARMA vehicle leaves the platoon, it remains connected to the platform and receives from CARMA platform the speed profile that minimises fuel consumption for the rest of the journey until it reaches its highway exit or the driver requests to merge to another platoon. It is noted that, as the CARMA platform continuously monitors all the connected vehicles, if a CARMA vehicle or platoon is not able to follow its speed profile, e.g. due to traffic-jams, adverse road conditions, etc. the speed profile and/or route of the CARMA vehicle/platoon can be recomputed to accommodate the surrounding environmental changes.

[1]For instance in [12], for merging form the back, the relative distance is reduced form 300 m to 15 m.

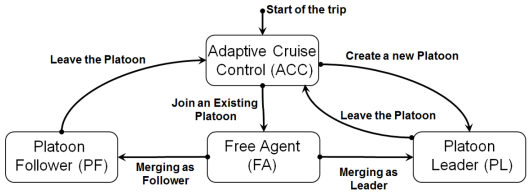

Fig. 3. Operation mode of a CARMA vehicle during a highway trip.

The operating mode for a CARMA vehicle in the scenario described above are depicted in Figure 3 which also shows the events determining the changes for one mode to another.

IV. System Architecture

The logical architecture for the platoon control and management and its distribution in the CARMA platform is shown in Figure 4. The architecture is composed by three layers, i.e. *Trip-Planner*, *Road Section Manager*, and *Coordination Control*.

The top layer, Trip-Planner computes the global optimal route and speed profile for the minimisation of the energy consumption on each section of the highway system and for all CARMA vehicles operating in any vehicle mode (ACC-mode, FA-mode and PL/PF-mode). For vehicle operating in platoons the Trip-Planner also provides a suggested inter-vehicular distance. To perform these tasks the Trip-Planner requires information with a spatial horizon that stretches all the way to the vehicles' destination, i.e. profile of the altitude for each highway segment, traffic condition, weather condition, road condition, etc. [18]–[20]. It is noted that algorithms for searching global optimum energy consumption are not easily tractable, because of their complexity in terms of computational burden which increases exponentially with the number of control variables and states. This drawback is known in the optimisation literature as "curse of dimensionality" [21], [22]. The need to access to information collected from a wide area and high computation demand for computing global optimal speed profiles suggest that Trip-Planner is implemented in CARMA Core Cloud as depicted in Figure 4. Notice that the use of cloud computing to solve global optimisation problem is also in accordance with the current literature in vehicle energy management systems [3], [18], [19], [23]. Usually, simplifying assumptions are used to solve optimisation problems, e.g., simple vehicle fuel consumption models, absence of traffic or simplified traffic models etc.

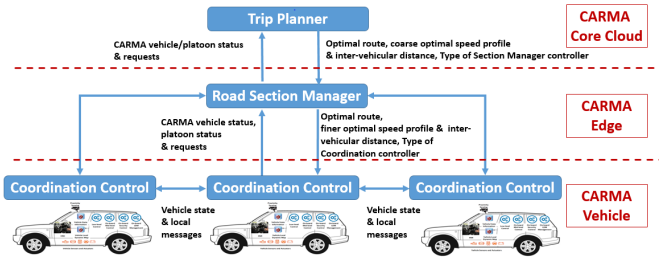

Fig. 4. Logical architecture for the platoon control and management.

Consequently, the Trip-Planner provides coarse references for the vehicles speed and inter-vehicular platoon distance. The Road Section Manager (RSM) is responsible for refining vehicle local speed profiles and inter-vehicular distance (for platoons already formed) in order to adapt them to the current road section state, such as road condition, real time traffic data, average vehicle speed on that section, length of the vehicle queue ahead etc. [17]. Furthermore, the RSM can also modify locally the speed profile of CARMA-vehicles in FA-mode and the speed of the corresponding target platoons to guarantee merging (i.e., the control algorithms for the pre-merging phase introduced in Section III is implemented in the RSM). The RSM is also used to support vehicle platoons by orchestrating activation of cooperative manoeuvring control algorithms running in the lowest layer of the architecture (i.e., the Coordination Control layer) and provide to them parameters for their correct planning and execution. It is noted that, when platoons are implemented through the sole use of V2V communication systems, platoon control tasks are usually coordinated by the platoon leader (i.e., centralised platoon management [9]). Followers take orders and send requests from/to the platoon leader. In so doing, only the platoon leader stores and manages sensible platoon data (e.g., list of the vehicle in the platoon, vehicle destination, vehicle acceleration/deceleration capability), thus enhancing privacy in situations where followers should not have access to the platoon configuration [16]. However, for centralised platoon management, the leader must have access to information that followers might not want to share (e.g. their destination or vehicle models for sorting vehicles in the platoon). Furthermore, information is sent from one vehicle to another any time the leader vehicle leaves the platoon, thus reducing the effectiveness of the centralised approach to preserve privacy of the platoon members. The use of the RSM in the proposed architecture can mitigate such a drawback as all the sensible data of the platoon are stored and used in the RSM, thereby preventing the information sharing to any platoon vehicles. It is noted that to perform tasks within the RSM, real-time traffic data needs to be provided. Furthermore, latency for providing the speed trajectories to the connected vehicle should be small to avoid loss of performance, i.e., unexpected increase of the fuel consumption due to a delay to compute and actuate the optimal speed profiles [19]. For these reasons, the RSM is suggested to be implemented within the CARMA Edge as it can provide required communication latencies through the use of 5G communication technologies and a better description of the surrounding traffic by exploiting real-time traffic data collected from all vehicles sharing the road section. In the lowest level of the architecture, i.e., Coordination Control, there are controllers for imposing vehicle speed profiles, implementing cooperative adaptive cruise control methods (platooning), and planning/executing merging and leaving manoeuvres. It is noted that, to safely perform merging, platooning and leaving operations, a tight coordination of the movements of the platoon vehicles is required [1]. Such synchronised movements are achieved through feedback control systems based on real-time data of the states of the platoon vehicles (i.e. position, velocity and acceleration), thus, a sufficiently small communication latency for vehicle data exchanges must be guaranteed [24]. Furthermore, as the control variable of such control systems (e.g., demanded vehicle acceleration and steering) directly affect the vehicle motion, for safety reasons they must be implemented on-board, thus they are sited in the CARMA vehicle layer of the architecture in Figure 1. It is remarked that in Figure 4, the horizontal arrows among CARMA vehicles represent data and messages (e.g. commands and acknowledging for operations completed [16]) exchanged among vehicles of a platoon for successfully completing cooperative manoeuvrings and cooperative driving. Vehicles in a platoon might exchange such data and messages through CARMA edges, i.e., by continuously uploading/downloading such information to/from the CARMA edges. However, if the resulting latency is too large to guarantee platoon stability or communication with the CARMA edges is lost, a V2V communication system might be adopted and integrated in the CARMA architecture to ensure an acceptable communication delays.

A. *Platoon Architecture: Trip-Planner Block*

The functionality modules of Trip-Planner block are shown in Figure 5. The block named CORE Vehicle Manager (CVM) accesses and updates data for the correct management of the vehicles and platoons. At the cloud level, each vehicle connected to the CARMA is characterised by a set of information such as its identifier (ID), vehicle destination, optimal route and a course optimal speed profile, current vehicle position, macro vehicle state (e.g., ACC, FA, PF and PL mode), etc. Similarly, each platoon in the CARMA platform is characterised by a platoon ID, ID of the leader

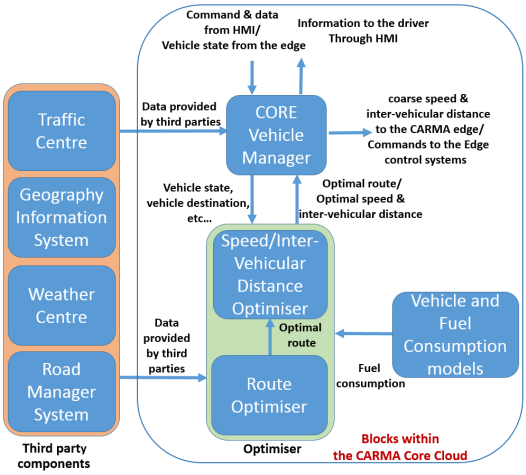

Fig. 5. Trip-Planner: fundametal blocks.

and followers, actual number of vehicle in the platoon, maximum length of the platoon that can preserve stability, a course optimal speed and inter-vehicular profiles, etc. The information is provided as input to the CVM by the driver through the on-board HMI (such as destination, platoon to join, etc.), CARMA Edges (e.g. current vehicle/platoon location and speed), and other modules implemented within the CARMA Core cloud (e.g. optimal vehicle route, optimal vehicle/platoon speed profile etc.). The outputs of the CVM block are information to the driver (e.g., the list of possible platoons on the vehicle route) and to the RSM layer (e.g., optimal route, speed profile to be refined, the controller to be selected, etc.). For the computation of the optimal route, optimal vehicle/platoon speed profile and optimal platoon inter-vehicular distance the CVM is supported by the Optimiser block as shown in Figure 5. The events at which CVM triggers the optimiser block can be for instance: (i) when the vehicle enters in the highway system and it is in ACC mode where the Optimiser is executed to compute the optimal route of the vehicle and its speed profile over the entire trip; (ii) when a vehicle in FA mode has completed the merging manoeuvre to join a platoon where the Optimiser is executed to compute the optimal speed profile and inter-vehicular distance for that platoon from the merging point to the next leaving point, or (iii) when a vehicle in PF and PL mode has completed the leaving manoeuvre to exit a platoon where the Optimiser is executed to compute the optimal speed profile and inter-vehicular distance for that platoon until the next leaving point and the optimal speed profile of vehicle exiting the platoon until its exit gate. Furthermore, to consider the variability of the surrounding conditions (e.g., traffic, road condition etc.), the CVM triggers the optimisation block if the vehicles and platoons connected to the CARMA platform do not follow the optimal speed plan or if the discrepancy among the actual speed profile and the optimal one is above a given threshold. The Optimiser block is composed by two sub-modules, i.e. Route Optimiser and Speed/Inter-Vehicular Distance Optimiser. The Route Optimiser module computes the path on the highway system from the entry gate to the exit gate that minimises a criterion (e.g., length of the path, travel time etc.). Instead, the Speed/Inter-Vehicular Distance Optimiser computes references for single vehicle and platoons connected to the CARMA platform with the aim to minimise fuel consumption. For vehicles operated in the ACC and FA modes, this block computes the most energy efficient speed profiles. For vehicles operated in platoons, in addition to the reference platoon speed, the block provides also the best inter-vehicular distance to further reduce the platoon fuel consumption [23]. It is noted that, the knowledge of the vehicle and fuel consumption models are fundamental to compute feasible speed profiles and estimate the fuel consumption [3], [18], [19], [23]. In the architecture shown in Figure 5, these models are provided by the Vehicle and Fuel Consumption models block. Besides, it is remarked that the accuracy of the optimisation procedures for the minimisation of the fuel consumption can be further enhanced if traffic data and upcoming road parameters (e.g., road surface friction, maximum speed velocity, variation of the altitude along the road etc.) are known [25]. In the architecture in Figure 5, and in agreement with the general CARMA architecture, this additional data is provided by third parties and is collected in the block Third Party Components block which include (i) the traffic centre block to provide current and forecasted traffic data; (ii) the Geography Information System block to provide the altitude profile over the selected optimal path; (iii) Weather centre block which provides weather forecasts for the computation of the road condition and (iv) the Road Manager System block which provides additional information about segments of the highways system such as unavailable road segments, status of the road [6], etc.

B. Platoon Architecture: Road Section Manager

The architecture for the RSM is depicted in Figure 6. The main components are the EDGE Vehicle Manager (EVM) and the Speed Planner (SP) blocks. The EVM block accesses and updates data for the correct management of the vehicles and platoons like those managed by the CVM module in Section IV-A. However, it retains only data for the management of the highway section covered by the CARMA Edge where it is implemented. Also the EVM block communicates with the Coordination Control layer, and provides to each vehicle connected to the CARMA platform the speed profile that must be actuated while travelling

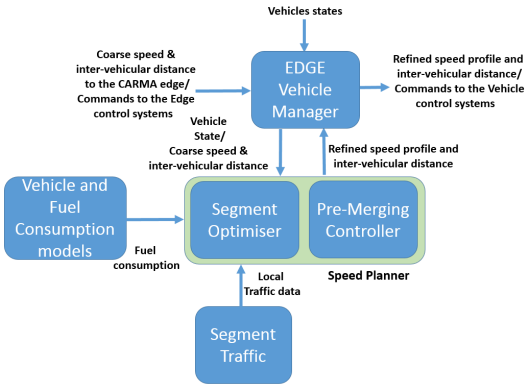

Fig. 6. Road Section Manager: fundametal blocks.

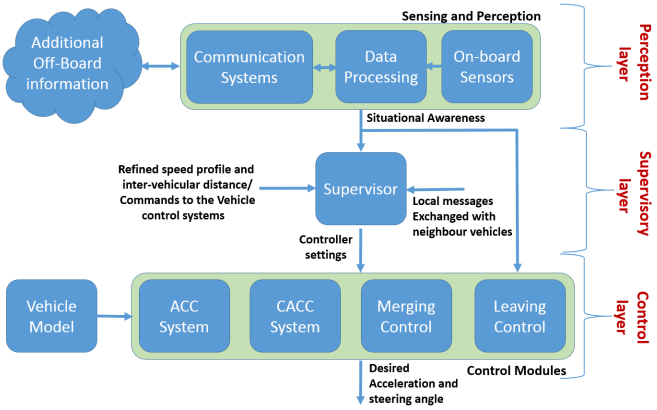

Fig. 7. Coordination Control: fundametal blocks.

that highway section. In addition, for vehicles organised in platoons, the EVM block provides also the inter-vehicular distance to be maintained. Finally, the EVM module sends commands to the underlying level of the architecture for the orchestration of the platoon manoeuvres, e.g. the start of the merging manoeuvre when the free agent vehicle is in the right position to merge in its target platoon. Time for the generation of such commands, e.g. start merging/exit to/from the platoon, can be computed by the EVM as this block continuously monitor the state of the vehicles and platoons connected to the CARMA platform and compare them to the travel plan of each vehicle. It is noted that, in the case of a side merging, the EVM also decides the position of the new platoon member in the string in accordance to some criteria which is not revealed to all platoon members for privacy reasons. The speed profile provided to vehicles and platoons is computed by SP block which is composed by two sub-modules, Segment Optimiser and Pre-Merging controller. According to the vision described in Section 2, on a highway section, there are mainly three categories of vehicles connected to the CARMA platform, i.e., vehicle in ACC mode, vehicles organised in platoons, and vehicles in free agent mode travelling on the same road section of their target platoon. For the first two categories, the SP block uses the algorithms implemented in the Segment Optimiser to generate speed variations around speed profile given by the upper layer for further reducing fuel consumption on the road section [26]. To this aim, in addition to vehicle dynamics and fuel consumption models, the Segment Optimiser uses real-time local traffic information, such as the average speed on the segment or the length of vehicle queue ahead [17]. Such local traffic information is provided to the optimiser by the Segment Traffic block. In the case a free agent and its target platoon share the same road segment, the SP block uses control algorithms within the sub-block Pre-Merging controller to modify their speeds so as to create a point over the section where the merging phase can initiate [12].

C. Platoon Architecture: Coordination Control

The architecture for the Coordination Control block of Figure 4 is shown in Figure 7. Similar to the architecture proposed in [13], three layers can be identified, e.g., Perception layer, Supervisory layer and Control Layers within Coordination Control block. The Perception layer provides data describing the surrounding environment of a CARMA vehicles, such as localisation of the subject vehicle and adjacent vehicle (including position of the vehicle in the target platoon) on a digital map, lane markers, road condition etc. It is noted that, in addition to on-board sensory information from such as Camera, radar, Lidar etc., off-board data can be used to improve the accuracy of the knowledge of the surrounding environment by enabling cooperative perception [27] (see for instance [28] for a detailed review about cooperative localisation through V2X communication systems). The control layer contains the control modules to generate vehicle accelerations and steering to execute platoon manoeuvres (i.e. merging and leaving platoons) and control modules for maintaining the platoon (i.e. modules for actuating CACC). In addition an ACC module is included in the control layer to control vehicle speed for vehicles in ACC and FA mode. The activation/deactivation of the control modules are orchestrated by the Supervisor block within Supervisory layer, which also provides to the underlying control layer control settings, such as required speeds and inter-vehicular distances, lane to move for a side platoon merging, etc. for the correct functioning of feedback control algorithms of the Control Modules. To this aim, the Supervisor communicate both with the EVM and the other vehicles. From the EVM, it receives the vehicle/platoon speed, required inter-vehicular distance, and high level commands, such as the command

enabling the switching from the ACC System to Merging Control. Vehicles in a platoon communicate to exchange vehicle states and messages for the correct execution of platoon manoeuvres. These messages can be commands (such as creating additional gap between consecutive vehicles for a side merging [13]) or acknowledge when a command has been executed.

V. Conclusion

This paper has presented a three layer functional platoon architecture to be potentially assisted by the cloud computing framework under development within the CARMA project. The architecture has been designed for passenger vehicles and can be used in mixed traffic scenarios. The functionalities distributed in the upper and middle layers aim to support fuel efficient control and management of vehicles and platoons within the systems, while the lowest layer is used to actuate commands and local platoon manoeuvres. Future work on the platoon will investigate the functionalities to be implemented in detail and propose fault tolerant distributed control systems that take advantage of the CARMA platform.

Acknowledgment

This work was supported by Jaguar Land Rover and the UK-EPSRC grant EP/N01300X/1 as part of the jointly funded Towards Autonomy: Smart and Connected Control (TASCC) Programme and is subject to UK patent under UKIPO GB1804663.1 (under review).

References

[1] U. Montanaro, S. Dixit, S. Fallah, M. Dianati, A. Stevens, D. Oxtoby, and A. Mouzakitis, "Towards connected autonomous driving: review of use-cases," *Vehicle System Dynamics, available online*, 2018.

[2] SARTRE, "Safe road trains for the environment; developing strategies and technologies to allow vehicle platoons to operate on normal public highways," 2012. [Online]. Available: http://www.sartre-project.eu

[3] S. Eilers, J. Martensson, H. Pettersson, M. Pillado, D. Gallegos *et al.*, "COMPANION-Towards Co-operative Platoon Management of Heavy-Duty Vehicles," *IEEE Conference on Intelligent Transportation Systems*, pp. 1267–1273, 2015.

[4] S. Tsugawa, "An Overview on an Automated Truck Platoon within the Energy ITS Project," *IFAC Symposium on Advances in Automotive Control*, pp. 41–46, 2013.

[5] J. Ploeg, A. F. A. Serrarens, and G. J. Heijenk, "Connect & Drive: design and evaluation of cooperative adaptive cruise control for congestion reduction," *Journal of Modern Transportation*, vol. 19, no. 3, pp. 207–213, 2011.

[6] Volvo, "Volvo Cars puts 1000 test cars to use: Scandinavian cloud-based project for sharing road-condition information becomes a reality," 2014. [Online]. Available: https://www.media.volvocars.com

[7] P. Varaiya and S. Shladover, "Sketch of an IVHS systems architecture," *Vehicle Navigation and Information Systems Conference, 1991*, vol. 2, pp. 909–922, 1991.

[8] S. Tsugawa, S. Kato, T. Matsui, H. Naganawa, and H. Fujii, "An architecture for cooperative driving of automated vehicles," *IEEE Intelligent Transportation Systems*, pp. 422–427, 2000.

[9] S. Halle, J. Laumonier, and B. Chaib-Draa, "A decentralized approach to collaborative driving coordination," *IEEE International Conference on Intelligent Transportation Systems*, pp. 453–458, 2004.

[10] T. Litman, "Autonomous Vehicle Implementation Predictions: Implications for Transport Planning," *Transportation Research Board Annual Meeting*, vol. 42, pp. 36–42, 2014.

[11] A. Stevens, M. Dianati, K. Katsaros, C. Han, S. Fallah, C. Maple, F. McCullough, and A. Mouzakitis, "Cooperative automation through the cloud: The CARMA project," in *ITS European Congress*, 2017.

[12] G. M. A. Arnaout and J.-P. C. Arnaout, "Exploring the effects of cooperative adaptive cruise control on highway traffic flow using microscopic traffic simulation," *Transportation Planning and Technology*, vol. 37, no. 2, pp. 186–199, 2014.

[13] E. S. Kazerooni and J. Ploeg, "Interaction Protocols for Cooperative Merging and Lane Reduction Scenarios," *IEEE Conference on Intelligent Transportation Systems*, pp. 1964–1970, 2015.

[14] R. Hall and C. Chin, "Vehicle sorting for platoon formation: Impacts on highway entry and throughput," *Transportation Research Part C: Emerging Technologies*, vol. 13, no. 5-6, pp. 405–420, 2005.

[15] A. Mihály and P. Gáspár, "Control of platoons containing diverse vehicles with the consideration of delays and disturbances," *Transportation Engineering*, vol. 40, no. 1, pp. 21–26, 2013.

[16] M. Amoozadeh, A. Raghuramu, C. N. Chuah, D. Ghosal, H. Michael Zhang, J. Rowe, and K. Levitt, "Security vulnerabilities of connected vehicle streams and their impact on cooperative driving," *IEEE Communications Magazine*, vol. 53, no. 6, pp. 126–132, 2015.

[17] I. Johansson, J. Jin, X. Ma, and H. Pettersson, "Look-ahead speed planning for heavy-duty vehicle platoons using traffic information," *Transportation Research Procedia*, vol. 22, pp. 561–569, 2016.

[18] E. Ozatay, S. Onori, J. Wollaeger, U. Ozguner, G. Rizzoni, D. Filev, J. Michelini, and S. Di Cairano, "Cloud-based velocity profile optimization for everyday driving: A dynamic-programming-based solution," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 6, pp. 2491–2505, 2014.

[19] C. Qiu, H. Shen, A. Sarker, V. Soundararaj, M. Devine, A. Rindos, and E. Ford, "Towards green transportation?: Fast vehicle velocity optimization for fuel efficiency," in *International Conference on Cloud Computing Technology and Science*, pp. 59–66.

[20] C. Sun, S. J. Moura, X. Hu, J. K. Hedrick, and F. Sun, "Dynamic Traffic Feedback Data Enabled Energy Management in Plug-in Hybrid Electric Vehicles," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 3, pp. 1075–1086, 2015.

[21] M. Ansarey, M. Shariat Panahi, H. Ziarati, and M. Mahjoob, "Optimal energy management in a dual-storage fuel-cell hybrid vehicle using multi-dimensional dynamic programming," *Journal of Power Sources*, vol. 250, pp. 359–371, 2014.

[22] R. Bellman, *Dynamic Programming*. Princeton, 1957.

[23] N. Murgovski, B. Egardt, and M. Nilsson, "Cooperative energy management of automated vehicles," *Control Engineering Practice*, vol. 57, pp. 84–98, 2016.

[24] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, "Vehicular Networking: A Survey and Tutorial on Requirements, Architectures, Challenges, Standards and Solutions," pp. 1–33, 2011.

[25] C. Zhang, "Predictive Energy Management in Connected Vehicles: Utilizing Route Information Preview for Energy Saving," Ph.D. dissertation, Clemson University, 2010.

[26] H. Lim, W. Su, and C. C. MI, "Distance-based ecological driving scheme using a two-stage hierarchy for long-term optimization and short term adaptation," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, 2017.

[27] S.-W. Kim, W. Liu, M. H. Ang, E. Frazzoli, and D. Rus, "The Impact of Cooperative Perception on Decision Making and Planning of Autonomous Vehicles," *IEEE Intelligent Transportation Systems Magazine*, vol. 7, no. 3, pp. 39–50, 2015.

[28] S. Kuutti, S. Fallah, K. Katsaros, M. Dianati, F. Mccullough, and A. Mouzakitis, "A survey of the state-of-the-art localisation techniques and their potentials for autonomous vehicle applications," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 829 – 846, 2018.