V2X Congestion Control for Multi-Channel Operation: a Scalable Validation in Virtualized Environments

Miguel Sepulcre[1], Yeray Guadalcazar[1], Miguel A. Fornell[2], Gokulnath Thandavarayan[1,3],
Francisco Paredes Vera[2], Javier Gozalvez[1], Amir Mohammadisarab[1]
[1]Uwicore lab., Universidad Miguel Hernández de Elche (UMH), Spain.
Email: {msepulcre, yeray.guadalcazar, gthandavarayan, j.gozalvez, amohammadisarab}@umh.es
[2]Idneo Technologies, SAU, Barcelona, Spain. E-mail: {miguel.fornell, francisco.paredes}@idneo.com
[3]Shiv Nadar University, Chennai 603110, India. E-mail: gokulnatht@snuchennai.edu.in

***Abstract*—Connected and automated driving introduces a myriad of new V2X services, such as cooperative perception and maneuver coordination, that significantly increase the channel load and require multi-channel V2X operation. Policies for Multi-Channel Operation (MCO) and congestion control are therefore essential for simultaneously supporting multiple V2X services across several channels. In this context, this paper presents the design, implementation, and extensive validation of a Facilities layer V2X congestion control solution for multi-channel operation integrated into an ETSI-compliant Cooperative Intelligent Transportation Systems (C-ITS) protocol stack. Our approach dynamically adapts transmission parameters based on real-time channel conditions and the priorities and requirements of the V2X services operating in a C-ITS station. By employing a Traffic-Class based proportional fairness strategy, the solution allocates available communication resources among multiple V2X services, effectively responding to varying channel loads in real time. Scalable experimental results in a virtualized environment demonstrate that our solution meets ETSI Release 2 requirements while bridging the gap between simulation-based evaluations and real-world testing, accounting for hardware limitations and processing delays. This work lays a robust foundation for scalable and congestion-aware C-ITS testing and validation prior to real world deployments. This paper makes our code publicly available so that other researchers can replicate our study and further explore MCO solutions for V2X communications.**

Keywords- Congestion Control, Multi-channel operation, MCO, Facilities Layer, Vanetza, C-ITS stack, connected and automated vehicles, DCC, ETSI, CAV, V2X, vehicular networks, cooperative ITS, virtual environment.

I. INTRODUCTION

Vehicle-to-Everything (V2X) communications play a crucial role in enhancing road safety, traffic efficiency, and automated driving by enabling real-time information exchange among vehicles, infrastructure, and other road users. The ETSI Technical Committee on Intelligent Transport Systems (ITS) is responsible for developing standards for V2X communications, ensuring interoperability and addressing emerging vehicular communication needs. The initial set of standards, known as Release 1, was designed to support fundamental "Day-1" applications, which primarily relied on the exchange of Cooperative Awareness Messages (CAMs) [1] and Decentralized Environmental Notification Messages (DENMs) [2]. CAMs are continuously broadcast by vehicles and roadside units to provide real-time information about their status (e.g. position and speed in the case of vehicles). DENMs are event-driven messages intended to warn road users of safety-critical incidents. A single 10-MHz radio channel was deemed sufficient for "Day-1" applications, and Release 1 standards did not include mechanisms for multi-channel operation. To avoid overloading the channel, a framework for congestion control (DCC, Decentralized Congestion Control) was specified for ITS-G5. For C-V2X, the congestion control approach defined by 3GPP at the MAC was adopted in ETSI specifications.

The evolution of vehicular applications beyond "Day-1" functionalities has required a more advanced communication framework. ETSI Release 2 introduces a broader set of use cases, enabling vehicles to share collective perception messages [3] and maneuver coordination messages [4], allowing road-infrastructure nodes to perform automated vehicle marshaling for parking or factories [5][6], and enabling vulnerable road users (VRUs), such as cyclists and scooter riders, broadcast their presence [7]. These new V2X services significantly increase the volume and complexity of exchanged messages, requiring more than one 10-MHz radio channel [8]. To address this growing demand, ETSI Release 2 has defined a framework for multi-channel operation (MCO) through a set of specifications dedicated to managing the radio channels [9]. ETSI defines in [10] the communication architecture for MCO, specifying how different MCO entities within a C-ITS station gather information and dynamically make decisions about channel usage to ensure efficient and coordinated spectrum utilization. A key element of the MCO architecture is the new facilities-layer entity (a.k.a. MCO_FAC) [11], which is responsible for controlling and distributing the load among the available channels. MCO_FAC collects information about active V2X services and their communication requirements, monitors available radio access technologies, and dynamically allocates resources to active V2X services to optimize spectrum utilization.

The first studies on the ETSI MCO concept have been based on simulations, focusing on evaluating different mechanisms for distributing the load across the available channels [12][13]. In this paper, we present for the first time a real-world implementation of MCO_FAC on an ETSI-compliant C-ITS protocol stack, Vanetza, and its real-time evaluation under large-scale scenarios using virtualized environments. Our goal is to demonstrate the feasibility and potential of MCO_FAC when integrated into a C-ITS stack with multiple V2X services. Unlike simulation studies, the evaluation of our implementation accounts for potential hardware limitations and processing delays. Our work thus demonstrates a clear transition from simulation-based evaluations to real-world deployment, and provides valuable

This work is partially funded by *Centro para el Desarrollo Tecnológico y la Innovación* (CDTI) through the InPercept project Ref. PTAS-20211011. InPercept project is also supported by Spanish *Ministerio de Ciencia e Innovación* and receives funds from NextGenerationUE program. This work has also been partly funded by MCIN/AEI/10.13039/ 501100011033 and the "European Union NextGenerationEU/PRTR" (TED2021-130436B-I00).

insights into the dynamic resource allocation necessary to support the increasing complexity and communication requirements of V2X services. This real-time evaluation not only validates the theoretical benefits of MCO but also paves the way for more robust and scalable vehicular communication systems, ultimately contributing to safer and more efficient transportation networks.

With this paper we make our code open source [14], including extensions to Vanetza that implement MCO_FAC and integrate it seamlessly into the existing C-ITS protocol stack. In addition, we provide a complete multi-V2X service testing environment with all the necessary scripts and code to virtualize and run multiple concurrent instances of the C-ITS protocol stack. The provided source code allows other researchers to easily replicate our study and further explore MCO solutions for V2X communications.

II. MCO AT FACILITIES LAYER

The MCO architecture [10] supports multiple applications and services, as well as multiple radio interfaces and channels. The MCO architecture comprises three core components [9]: MCO_FAC (Facilities layer), MCO_NET (Network layer), and MCO_ACC (Access layer). MCO_FAC serves as the core decision-making component, where all the intelligence of the MCO operations reside, ensuring that channel selection strategies are dynamically adapted to meet the requirements of the different V2X services within C-ITS station. Its central role emerges from the need to consider both application requirements and access layer conditions, thereby facilitating efficient and coordinated use of multiple communication channels. ETSI defines in [11] the specific functionalities required to support MCO at the Facilities layer, ensuring seamless integration with V2X services as well as lower-layer networking and access technologies. For this purpose, the MCO_FAC contains the following three key entities:

Bandwidth Management Entity (BME). This entity computes and distributes the available communication resources among the V2X services of the C-ITS station. For this purpose, it monitors the load on each channel (via the corresponding radio interface) and computes the communication resources that the C-ITS can use. It then allocates the computed resources among the V2X services while taking into account their requirements. The resources can be internally computed as a proportion of the bandwidth or channel, but must be translated to a metric that is understandable by the V2X services, such as bits per second, so that they adapt their message rate and size as needed. The V2X services have the flexibility to adjust their message rate while keeping the message size constant (e.g., modifying the CAM generation without changing its size). They could also choose to modify the message size while maintaining the message rate (e.g., varying the number of detected objects included in each CPM), or they could opt to adjust both parameters (e.g., reducing the message size by omitting optional elements while also altering the message rate).

Message Handling Entity (MHE). This entity manages the transmission of messages to the Networking & Transport layer, ensuring that each message is assigned appropriate parameters (e.g., priority, transmission channel) based on the available communication resources. The MHE also collects statistics on the messages generated by the various V2X services to ensure that they comply with the limits indicated by the BME in terms of resources. These limits can be enforced individually for each service, or by aggregating the messages from all services.

Message Collecting Entity (MCE). This entity handles the reception of messages from lower layers and directs them to the appropriate V2X service. It can also collect statistics to enable advanced functionalities.

Although the MCO framework specifies the functional structure of MCO_FAC, the implementation of specific algorithms is left to manufacturers. This is particularly true, and relevant, for the algorithms used to compute the communication resources available to the C-ITS station in a channel based on its channel load, and to distribute those resources among the multiple active V2X services.

III. SYSTEM DESIGN AND IMPLEMENTATION

A. C-ITS protocol stack

For the prototyping of MCO_FAC, we have leveraged Vanetza [18], an open-source C-ITS protocol stack designed for vehicular networking research and development. Vanetza provides an implementation of the ETSI C-ITS protocol stack, enabling direct vehicle-to-vehicle and vehicle-to-infrastructure communication. It includes key functionalities such as GeoNetworking, the Basic Transport Protocol (BTP), and support for CAMs and DENMs in ASN1. Vanetza has been shown to successfully run on devices from Cohda Wireless (MK5), Autotalks (*Craton* and *Secton*), and *nfiniity* (CUBE EVK). We have also demonstrated in a previous study that Vanetza can run on an automotive grade Telematic Control Unit from Idneo (VMax) which is equipped with a C-V2X Mode 4 radio interface [19].

The MCO_FAC module implemented has been integrated into the Vanetza C-ITS protocol stack with minimal modifications. This development is based on *Socktapp*, a tool provided within Vanetza, which we have extended to support MCO functionalities at the Facilities layer. The implemented MCO_FAC is instantiated conditionally (via the *use-mco* flag) and can therefore be activated or deactivated at startup by the user. The modifications performed in *main.cpp*, BTP, and the corresponding facilities-layer services are minimal, ensuring that the MCO_FAC implementation does not disrupt existing functionalities and can be easily integrated into new V2X services.

The integration of the MCO_FAC module in the C-ITS architecture is illustrated in Fig. 1. As shown in the figure, the MCO_FAC module acts as an interface between different V2X services and the transport layer (i.e., BTP). The dashed line represents the control interface that handles the signal exchange for the proper interaction of the MCO_FAC with the V2X services, while the solid lines represent the data interfaces responsible for the exchange of data messages between different layers.

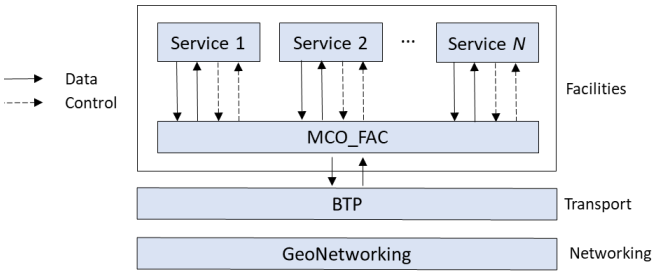


Fig. 1. Integration of MCO_FAC in the C-ITS protocol stack.

B. *Facilities layer MCO*

Fig. 2 illustrates the internal architecture and message flow of the implemented MCO_FAC module. The core component, which is part of the BME, performs resource management and is triggered periodically by a timer set to T_{RM} = 200ms. This component first computes the resources available to the C-ITS station based on the channel load, and then distributes these computed resources among the V2X services. To achieve this, we have implemented the Adaptive approach of DCC (Decentralized Congestion Control) [16] at the Facilities layer. This approach calculates the amount of resources that the C-ITS station can use, referred to as delta (δ), a unitless value representing the maximum percentage of time (or bandwidth) the station is allowed to transmit on the channel. δ is updated in each iteration using a control equation that depends on the target and the measured channel load:

$$\delta = (1 - \alpha) \cdot \delta + \delta_{offset} \quad (1)$$

where

$$\delta_{offset} = \begin{cases} \min(\beta \cdot (CBR_{tar} - CBR), G^{+}_{max}) & if\ CBR_{tar} > CBR \\ \max(\beta \cdot (CBR_{tar} - CBR), G^{-}_{max}) & if\ CBR_{tar} \leq CBR \end{cases} \quad (2)$$

α, β, G^{+}_{max} and G^{-}_{max} are constant control parameters [16]. The channel load is measured every T_{RM} using the CBR (Channel Busy Ratio) metric, which represents the proportion of time the channel is sensed as busy. The target channel load, CBR_{tar}, was configured to 0.68, following [16].

Once δ is computed, the resource management component adopts a Traffic-Class based proportional fairness approach to distribute the resources among the V2X services of the C-ITS station. This approach is inspired from [17] but the design of alternative approaches is an open research challenge, and actually the objective of the Special Task Force STF688 on Resource management that has been recently established within ETSI. Following this approach, the highest-priority services (i.e., those in the lowest Traffic Class) receive resources in proportion to their requirements. If any resources remain, they are then allocated to the second-highest Traffic Class, with any subsequent surplus distributed to the third and fourth Traffic Classes until all resources are assigned. To prevent any V2X service from being starved, we have implemented a policy ensuring that each service is allocated at least the resources required to transmit one packet every 2 seconds. To estimate the resources required by each V2X service, the MHE of MCO_FAC collects statistics on all generated messages. The data collected is then used to compute the average message size and transmission interval for each service over the last second, thereby determining the average required resources for each V2X service. Some of the key parameters of our implemented MCO_FAC are summarized in Table 1.

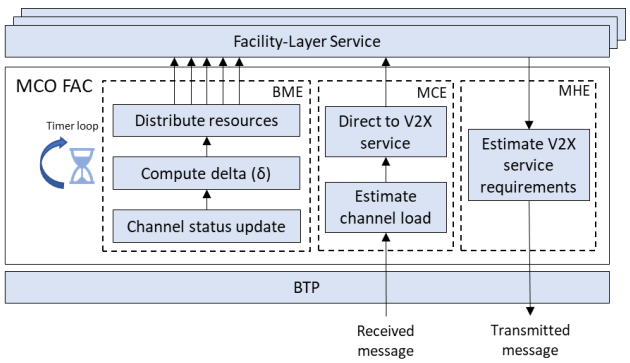


Fig. 2. Internal architecture and flow of the implemented MCO_FAC.

Table 1. MCO_FAC parameters

Parameter	Value
Target channel load (CBR_{tar})	0.68
Traffic Classes	0–3
Maximum transmission interval	2 s
Resource management interval (T_{RM})	200 ms
Averaging time window	1 second

IV. Virtualized Testing Environment

The congestion control module at facilities layer (MCO_FAC) has been tested and validated in a virtualized testing environment. The module is implemented in a complete and fully standard-compliant C-ITS protocol stack. To run concurrent instances of the C-ITS stack on a single computer, a Docker container image was created that includes everything needed to run the stack. Each stack instance was deployed as a separate container at runtime with the Docker Engine. All containers were connected via the Docker network, a virtual network that allows containers to communicate with each other and with the outside world.

The MCO_FAC module was evaluated under low, medium, and high channel load scenarios by concurrently running five, ten, and fifteen virtualized C-ITS stacks, respectively. We have tested the implementation with up to fifteen stacks concurrently running on the same hardware due to the computational workload generated by the concurrent stacks and the processing power of our computing platform[1]. Our implementation can concurrently run more C-ITS stacks with more powerful computing platforms.

Each stack represents a C-ITS station (i.e. vehicle) and has five active V2X services that periodically generate V2X messages with constant size (105 bytes including headers). As shown in Table 2, each service is configured with a default minimum interval of 100 ms (i.e. a maximum message rate of 10 Hz). The considered V2X services will maintain their message size and reduce their message generation rate following the instructions of MCO_FAC to control congestion. The excess rate would be offloaded to an alternative channel. To validate the effect of message priorities, different Traffic Classes were configured for each V2X service (see Table 2).

Table 2. V2X message priorities

V2X service ID	Traffic Class
1	0
2	1
3	2
4	2
5	3

Since the communication among the virtualized C-ITS stacks happens through a virtualized network, the CBR measurements had to be estimated at the higher layers. For this purpose, the MCO_FAC collects information about all received V2X messages through the MCE. The CBR is estimated by summing the sizes of all the received V2X

[1] The implemented MCO_FAC has been tested on Ubuntu 22.04.3 LTS (64-bit) running on an Asus TUF Gaming FX505DT laptop, equipped with an AMD Ryzen 7 3750H processor operating at 2.3 GHz, 16 GB of RAM, and an NVIDIA GeForce GTX 1650 GPU with 4 GB.

messages every T_{RM} and dividing the result by T_{RM} and a predefined data rate of 6 Mbps. A scaling factor, s_f, was used to emulate high channel loads without the need to run more than one hundred stacks in parallel to congest the channel. The environment considered limited the maximum number of simultaneous stacks that run in real time to fifteen. The scaling factor was configured as s_f=9 to generate channel loads of approximately 31%, 63%, and 94% for the corresponding low, medium, and high channel load scenarios.

The duration of each test was configured to be 300 seconds (i.e. 5 minutes), with the first 50 seconds considered as an initial transition period and not considered for evaluation. This initial period is required for the simultaneous activation of all the stacks at startup in the same hardware, which will not happen in a real implementation. The length of this period depends on both the algorithm and the time required to initialize all the Docker containers on the same hardware.

V. Results

To evaluate the implemented MCO_FAC module, we first analyze its convergence. To this end, Fig. 3 plots the time evolution of delta (δ), which is the percentage of radio resources that can be used by the C-ITS station. The value of δ , computed from the measured CBR, is used by MCO_FAC to distribute the resources among the V2X services. Since the message size is constant, δ is employed to adapt the message generation interval of the V2X services. The results clearly indicate that the system becomes stable after the initial transition period. As expected, δ is inversely proportional to the channel load, decreasing as the channel load increases. In the low channel load scenario, we observed that δ converges to approximately 2.74%. For the medium channel load scenario, δ stabilizes at around 0.66%, and for the high channel load scenario, δ converges to around 0.45%.

The convergence and stability of δ is associated to the convergence and stability of the CBR, which is illustrated in Fig. 4. As observed, MCO_FAC does not apply any congestion control measures in the low channel load scenario because the load remains significantly below the target level. In the medium channel load scenario, MCO_FAC begins to implement slight congestion control measures as the load approaches the target. Finally, in the high channel load scenario, MCO_FAC effectively controls the channel load and reduces the CBR from 94% (if all services were generating messages at 10 Hz) to approximately 62%.

It is important to highlight that the stability and convergence of δ and the CBR were achieved while running the solution concurrently in Docker containers in real time. Hardware limitations introduced processing delays that affect the actual transmission intervals. This occurs because a timer is dynamically configured at runtime based on the message interval computed by MCO_FAC for each V2X service. When the timer expires, a new message is generated and transmitted, the processing delays prevent the actual transmission intervals from exactly matching those configured by MCO_FAC. Fig. 5 illustrates the time evolution of the configured and actual transmission intervals for one V2X service in the low channel load scenario. Fig. 5a shows that the configured interval is 100 ms, given the low CBR, while Fig. 5b shows that the actual transmission interval slightly fluctuates around 100 ms. These deviations are mainly due to real-time processing delays (e.g., computing overhead, hardware limitations, etc.) and do not impact the overall stability and convergence of the implemented MCO_FAC, thereby demonstrating its robustness. Moreover, these deviations capture realistic processing and timing characteristics often omitted in V2X simulations, underscoring the importance of this study for analyzing real-world implementations where precise timing and responsiveness are critical.

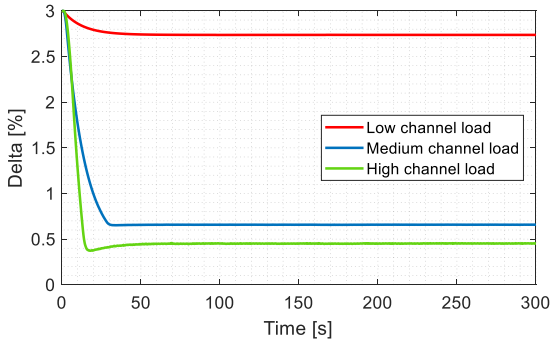


Fig. 3. Time evolution of delta (δ) for different channel load scenarios.

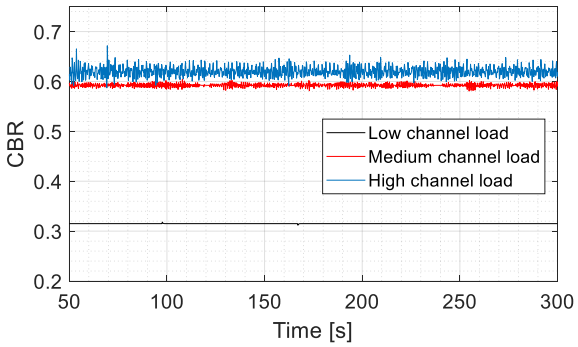
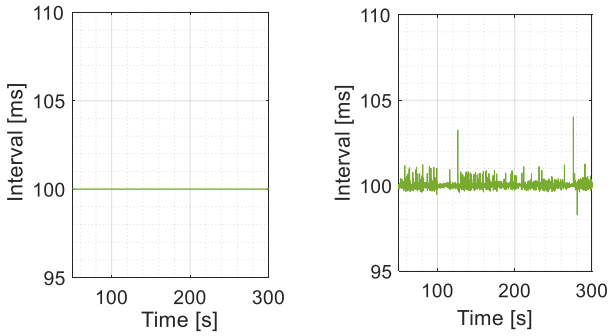


Fig. 4. Time evolution of CBR (Channel Busy Ratio) for different channel load scenarios.

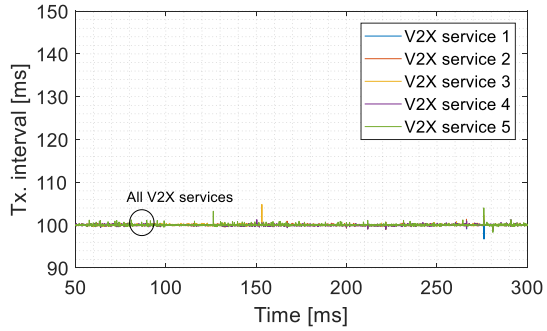


(a) Configured transmission interval (b) Actual transmission interval

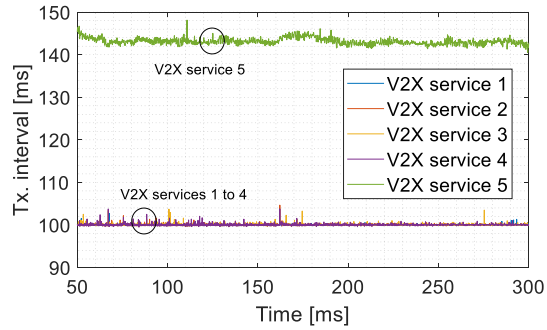
Fig. 5. Configured and actual message transmission interval of one V2X service in the low channel load scenario.

Fig. 6 shows the actual transmission intervals of all the V2X services as a function of time for the different channel load scenarios. Fig. 6a shows that the configured interval remains constant and is close to the default value of 100 ms in the low channel load scenario, primarily due to the low CBR (see Fig. 4). In contrast, for the medium and high channel load scenarios, the MCO_FAC limits the message rate of the V2X service with the lowest priority to control the channel load. Fig. 6b shows that the transmission interval of V2X service 5 increases to approximately 145 ms because there is insufficient bandwidth to accommodate all its transmissions. Conversely, in the high channel load scenario, MCO_FAC allocates all available resources to V2X services 1 to 4, as

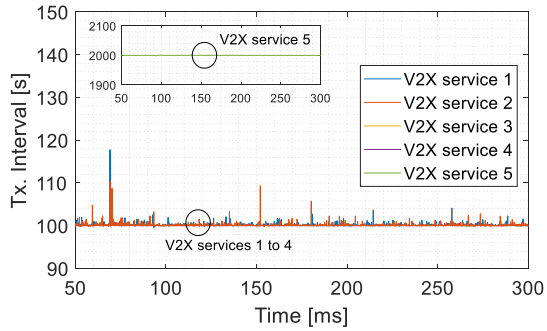
shown in Fig. 6c. To prevent starvation of V2X service 5, MCO_FAC still permits the transmission of one message every 2 seconds, following the implemented starvation policy.



(a) Low channel load scenario



(b) Medium channel load scenario



(c) High channel load scenario

Fig. 6. Actual transmission interval of V2X messages for different channel load scenarios.

VI. Conclusions

This paper has demonstrated the first successful design, integration, and scalable validation of a Facilities layer congestion control solution for multi-channel operation within an standards-compliant C-ITS protocol stack that is provided open-source to the community in [14]. The validation has been conducted in a virtualized experimental environment that allows testing the operation of congestion control protocols in scalable testing environments and real-world C-ITS stacks prior to on-field deployments. Our experimental evaluation confirms that the proposed solution dynamically adapts transmission parameters in response to real-time channel conditions and service priorities and requirements, effectively managing channel load even under hardware limitations and processing delays. The solution provides a practical framework that advances simulation-based evaluations toward real-world deployment. These findings establish a robust foundation for testing and validating scalable, congestion-aware C-ITS implementations. Future work will focus on designing and evaluating different algorithms for the resource management and message handling of MCO_FAC, integrating additional multi-channel operation components such as MCO_NET and MCO_ACC, and supporting ETSI V2X services such as the Collective Perception Service, the Maneuver Coordination Service or the Automated Vehicle Marshalling service.

References

[1] ETSI, “Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service”, EN 302 637-2, v1.3.2, 2014.

[2] ETSI, “Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service”, EN 302 637-3, v1.2.2, 2014.

[3] ETSI, “Intelligent Transport System (ITS); Vehicular Communications; Basic Set of Applications; Specification of the Collective Perception Service”, TS 103 324 V2.1.1, 2023.

[4] ETSI, “Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Maneuver Coordination Service”, TS 103 561, v0.0.8, 2024 (Draft).

[5] ETSI, “Intelligent Transport Systems (ITS); Automated Vehicle Marshalling (AVM); Release 2”, TS 103 882, v2.1.1, May 2024.

[6] F. A. Schiegg, et al, "Automated Vehicle Marshalling: The first Functionally Safe V2X Service for Connected Automated Driving", *IEEE Open Journal of Vehicular Technology*, Early Access, Feb. 2025.

[7] ETSI, “Intelligent Transport Systems (ITS); Vulnerable Road Users (VRU) awareness; Part 3: Specification of VRU awareness basic service; Release 2”, TS 103 300-3, v2.1.1, Nov. 2020.

[8] M. Sepulcre et al.,, "LTE-V2X Scalability and Spectrum Requirements to support Multiple V2X Services", Proc. *IEEE 100th Vehicular Technology Conference (VTC2024-Fall)*, Washington DC, USA, 7-10 Oct. 2023.

[9] A. Bazzi et al., “Multi-Channel Operation for the Release 2 of ETSI Cooperative Intelligent Transport Systems," *IEEE Communications Standards Magazine*, vol. 8, no. 1, pp. 28-35, March 2024.

[10] ETSI ITS, Intelligent Transport Systems (ITS); Architecture, Multi-Channel Operation (MCO) for Cooperative ITS (C-ITS)”, ETSI TS 103 697 V2.1.1, Nov 2022.

[11] ETSI ITS, Intelligent Transport Systems (ITS); Facilities Layer; Communication Congestion Control”, ETSI TS 103 141 V2.2.1, Nov 2022.

[12] E. Egea-Lopez and P. Pavon-Mariño, "Adjacent Channel Interference and Congestion Control for Multi-Channel Operation in Vehicular Networks," *IEEE Transactions on Intelligent Transportation Systems*, Early Access, Jan. 2025.

[13] O. Váczi, L. Bokor, “Modeling and Evaluation of a Dynamic Channel Selection Framework for Multi-Channel Operation in ITS-G5”, *Telecom*, vol. 4(2), pp. 313-333, 2023.

[14] Source code for V2X Congestion Control for Multi-Channel Operation over Vanetza C-ITS stack: https://github.com/msepulcre/mcoVanetza

[15] G. Thandavarayan, M. Sepulcre, J. Gozalvez and Baldomero Coll-Perales, “Scalable Cooperative Perception for Connected and Automated Driving”, *Journal of Network and Computer Applications*, vol. 216, 103655, July 2023.

[16] ETSI, “Intelligent Transport Systems (ITS); Decentralized Congestion Control Mechanisms for Intelligent Transport Systems operating in the 5 GHz range; Access layer part”, ETSI TS 102 687 V1.2.1, April 2018.

[17] ETSI, “Intelligent Transport Systems (ITS); Facilities Layer; Communication Congestion Control”, ETSI TS 103 141 V2.1.1, Nov 2021.

[18] Vanetza ETSI C-ITS protocol stack. Online: https://www.vanetza.org/ [Last access: Feb. 2025]

[19] S. Lopez, M. Sepulcre, M. Fornell, D. Quiñones, J. Espinosa, J. Gozálvez, P. Moga, "ETSI standard-compliant Collective Perception Service for Connected Automated Driving", *Proc. FISITA 2023 World Congress*, Barcelona (Spain), 12-15 September 2023.