ezCar2X: Rapid-Prototyping of Communication Technologies and Cooperative ITS Applications on Real Targets and Inside Simulation Environments

Karsten Roscher, Sebastian Bittl, Arturo A. Gonzalez, Matthias Myrtus, Josef Jiru

Fraunhofer ESK
80686 München
{karsten.roscher,sebastian.bittl,arturo.gonzalez,matthias.myrtus,josef.jiru}@esk.fraunhofer.de

Abstract

Vehicular networks and cooperative mobility are still active research fields. We present our approach to ease the design, implementation and testing of novel applications and protocols: the modular software framework ezCar2X. Cooperative ITS applications based on Car2X communication can be evaluated in a simulation environment and on real world targets with a single implementation. In this paper, we present the architecture and existing modules of the framework, describe the integration into a simulation environment and conclude with an outlook on planned improvements.

1 Introduction

In a world where everything is connected, Car2X communication is envisaged to be the next big step towards improved traffic safety and efficiency. Thus, most European car manufactures signed a Memorandum of Understanding promising a rollout of cooperative applications based on Car2X communication in Europe starting in 2015 [C2C12]. Active research in the last decade paved the way for the current set of ETSI standards – released or in the final stage of approval – that are required to ensure interoperability among different vendors. Research and development will continue driven by new applications and thus new requirements on the cooperative systems.

However, implementation and evaluation of new protocols and applications is still a demanding task. The design of distributed systems usually involves a combination of simulation as well as

real world prototyping which requires multiple implementations of the same algorithm in different environments. This is not only tedious but also error prone. Therefore, we present our approach: the modular software framework ezCar2X. With this framework, vehicle manufacturers, suppliers and road infrastructure operators can easily and quickly implement new applications based on Car2X communication and evaluate them in a simulation environment as well as on real world prototypes using a single implementation.

ezCar2X provides a basic set of standard compliant protocols, facilities and abstractions to cover a wide range of applications out of the box. All essential components can be replaced with updated or independently developed versions due to the framework's modular structure. Therefore, ezCar2X can also be a valuable tool in the research field, e.g. to evaluate new communication protocols in the context of existing applications.

The remainder of this paper is structured as follows. Section 2 discusses existing Car2X hard- and software solutions related to our work. The ezCar2X framework, its architecture and existing modules are presented in section 3. Based on the core concepts we describe the integration of the framework into a simulation environment in section 4. We conclude with an outlook in section 5.

2 Related Work

With many field operational tests concluded in recent years as well as the start of the commercial deployment of the so-called "Day 1" system on the horizon there exist a number of communication units, e.g. Denso WSU [Laso08], NEC LinkBird [Fest09] or Cohda Wireless MK [Gray09]. These units usually consist of the ETSI ITS-G5 compliant wireless hardware and a proprietary implementation of the associated communication protocols provided with a software development kit (SDK) to enable customers to implement their own applications. These applications can either run directly on the communication unit or on external hardware connected via Ethernet or USB. Since these units are targeted at commercial deployment the focus is on low costs, efficiency and stability. Implementations depend on the vendor specific libraries of the SDK and are thus not portable to other systems or environments. Furthermore, single protocols cannot easily be exchanged without re-implementing the whole stack due to the tight coupling of different layers. To the best of our knowledge, no solution exists that allows the usage of the target code also in a simulation environment apart from the common hardware-

in-the-loop (HIL) platforms. HIL however does not allow evaluating a larger number of systems with updated software at the same time.

In contrast, simulation provides the means to evaluate protocols and applications with many active devices under repeatable conditions. Communication systems are usually modeled using dedicated network simulators like ns-2 [Ns214], ns-3 [Ns314], OMNeT++ [OMNe14], Riverbed Modeler [Rive14] or JiST/SWANS [JiST14]. However, the availability of Car2X specific protocols and device models depends on the chosen tool. To the best of our knowledge, none of these tools provides a comprehensive model of the required ETSI standards for the physical, medium access, network and transport layers or these are at least not publicly available.

Furthermore, in the context of vehicular networks it does not suffice to consider only the communication system. Other aspects like road traffic, vehicular dynamics, sensor inputs and driver behavior have to be included as well. Road traffic including simple driver behavior models can be simulated with traffic simulators like the open-source SUMO [SUMO14] or the commercial VISSIM [VISS14]. Specialized tools, e.g. CarMaker [CarM14] or PreScan [PreS14], can be used to simulate the dynamics of a single vehicle in great detail. The combination of several simulators is a widespread approach, e.g. SUMO and ns-3 for the iTetris platform [iTet14] or SUMO and the OMNeT++ based MiXiM framework in Veins [Vein14], but it is often tied to a specific set of tools and therefore a limited scope of application since common interfaces to couple simulators do not exist. VSimRTI [VSim14] tries to overcome this problem introducing a flexible architecture for simulator coupling using an ambassador concept to ease the integration of new tools. Even though applications run in a distinct application simulator that allows reuse on a prototyping target, many protocols are still implemented directly in the respective simulators and are therefore not portable.

3 ezCar2X

ezCar2X is a modular software framework for rapid prototyping of cooperative ITS applications and novel communication protocols. A more detailed description as well as several real world application examples can be found in [RJGH13] and [SRJ13]. The interoperability of the access, network, security, and facility layer components with solutions provided by other parties was successfully demonstrated at the ETSI Plugtest event in 2013 [ETSI13b].

3.1 Architecture

The architecture of the framework is based on the European Telecommunication Standards Institute (ETSI) architecture for Intelligent Transport Systems (ITS) stations [ETSI10] with the respective access, network, facility, management and security layers as shown in Figure 1. The implementations of protocols and facilities meet recent standards or standard drafts and are kept up to date. Since most applications require data sources and actuators in addition to the communication system itself, additional components for seamless integration of in-vehicle bus systems, various sensors as well as digital maps are included. Each feature can be used separately or as part of a larger stack or framework.

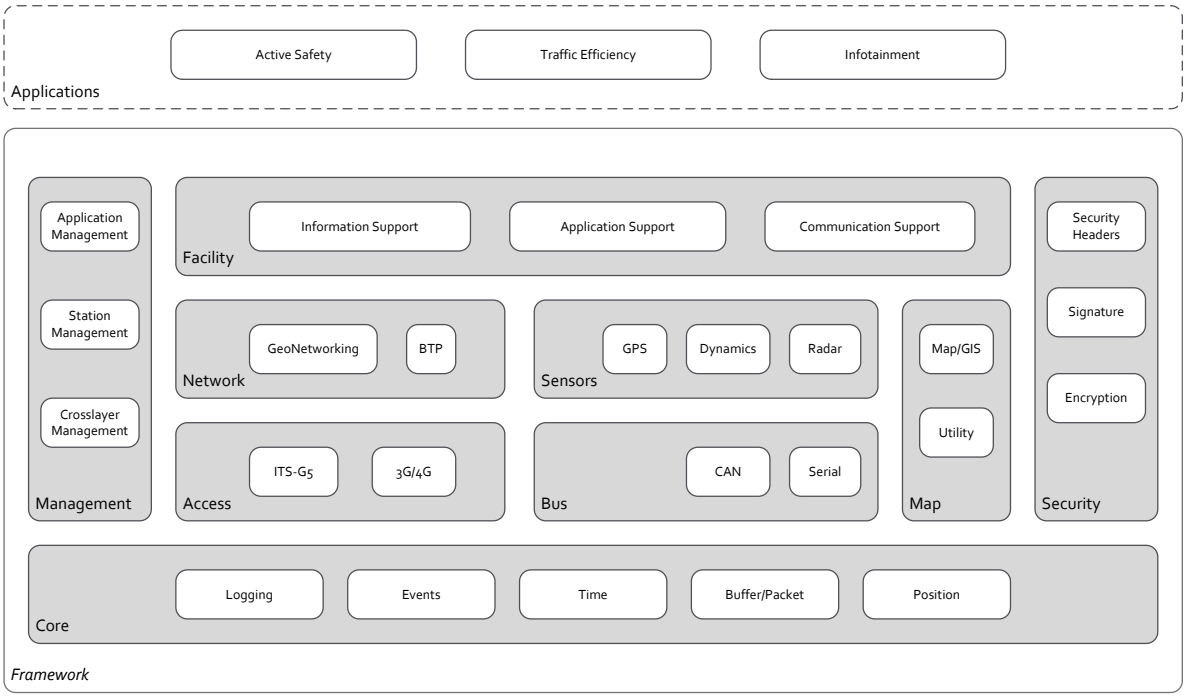

Fig. 1: ezCar2X framework architecture

ezCar2X is implemented in C++ leveraging platform specific optimization for an efficient use of system resources. Components are provided as a set of shared libraries with limited interdependencies. The basic principle to ensure both portability and modularity is abstraction. Each component is provided as a set of abstract base classes acting as interfaces. Thus, clients are decoupled from the actual implementation and different implementations can be used in different contexts without changes to existing code. Abstraction is not only used for plain software components but also to provide transparent access to external hardware such as wireless modems or field bus adapters.

Building a flexible framework out of individual components without a predefined structure is a challenging task. In ezCar2x single objects can be aggregated into a composite providing more complex functionality; an approach similar to the object aggregation mechanism in ns-3 [Ns314]. Specific features can be found by querying the entire aggregate for an interface that provides the requested feature. This allows for flexible "wiring" of different components as well as different cross-layer optimization techniques.

3.2 Modules

The *Core* module provides basic components and services shared among all other libraries. This includes a flexible logging system, an event scheduler for asynchronous tasks and simple timeout realization as well as buffer and data packet implementations with support for copy-on-write semantics. Furthermore, interfaces and data types related to time and position information are part of this module. Core also provides support for object aggregation, advanced object factories and an extensible property system for component configuration.

Support for ETSI ITS-G5 (with wrappers for various devices and platforms), GeoNetworking and the Basic Transport Protocol (BTP) according to ETSI standards is included in the *Access* and *Network* modules. Cellular services are supported as well. An adaptive GeoNetworking layer with support for an application transparent simultaneous use of local wireless communication and cellular connectivity is currently work in progress. The *Security* module provides an implementation of a network security entity for signing and encrypting messages to transmit as well as validating and decrypting received messages. Furthermore, it handles the management of available certificates and regular pseudonym updates.

The *Bus* module supports basic serial interfaces (RS232, USB) and Controller Area Networks (CAN). It is mostly used to integrate sensors and actuators within a vehicle or along the road. An additional abstraction layer is provided by the *Sensor* module to enable reuse of algorithms based on sensor input across different equipment types. The integration of a new sensor is simply a matter of implementing the interface of the specific sensor type. Several basic types, e.g. position, speed, acceleration and object detection (radar or laser scanner), are provided with the library. Furthermore, self-description of a specific sensor adaption provides properties, like accuracy, that vary among different devices of the same category. Data input is complemented by the *Map* module which provides abstract concepts to access digital map data as well as various routing and map matching algorithms. An input adapter for OpenStreetMap [OSM14] is provided with the framework and access to other map databases can easily be added.

The *Facility* library covers components common to most cooperative applications and is modeled according to the ETSI ITS architecture. It includes the Cooperative Awareness (CAM) [ETSI13a] and Decentralized Environmental Notification (DENM) [ETSI12] service, support for Signal Phase And Timing (SPAT) messages, status and position management, and support for data presentation and high level message encoding. In addition, integration of Human-Machine-Interfaces (HMI) is greatly simplified using an HMI server that manages all HMI devices connected via Bluetooth or any IP based connection. Information is exchanged via an event-based custom protocol that can easily be extended with new application data. Generic client implementations exist for Linux and Windows (both C++) and Android (Java).
Cross-layer information exchange and interaction is handled by the components in the *Management* module. Finally, the *Framework* module provides efficient tools to configure everything from a basic communication stack to a full-featured application unit using XML descriptions to select and configure the required components.

4 Towards Simulation Integration

Applications and protocols cannot always be evaluated to a full extent in real world test beds or field operational tests. Large scale deployments with hundreds or thousands of vehicles are expensive and hard to orchestrate. Thus, simulation plays a major role in the analysis of novel algorithms and approaches in cooperative systems. The flexible and modular architecture of ezCar2X makes it an ideal candidate to bridge the gap between simulation and rapid prototyping [BR14].
ezCar2X on its own is intended to provide a communication stack and other framework components for a single node within the network. However, the simulation of cooperative applications requires several nodes that exchange information. A basic version of such an environment is already provided by the framework itself. It consists of a virtual communication device for each stack and a dedicated server application to connect several instances of ezCar2X on a single machine. This approach can be used to implement and test applications without the need for actual wireless interfaces. However, it is not intended to resemble the behavior of real-world-hardware or provide the means to simulate more than a few ITS stations.

4.1 Simulation Environment

Realistic simulations require detailed models of the wireless interfaces and channels. Furthermore, the simple approach using virtual devices requires all instances to run in real-time using the system clock for synchronization. This leads to a severe limitation regarding the maximum number of nodes that can be simulated on a single machine. Therefore, we selected the open source network simulator ns-3 [Ns314] to overcome both of these issues. ns-3 already provides detailed implementations of devices of the IEEE 802.11-family as well as a variety of wireless channel models that are continuously extended and refined by the ns-3 community. Furthermore, ns-3 is based on the principle of discrete event simulation. Everything is reduced to an event assigned to a specific simulation time. Events can generate new events and are processed in the order of expiration. This leads to an execution time that is independent from real time; simulations can run faster or slower depending on the complexity of the models and the scenario. Another advantage of ns-3 in this context is its flexible structure implemented in C++. Since ezCar2X modules are provided as shared libraries they are also available to simulations written with ns-3. Thus, combining the two is only a matter of providing ns-3 specific implementations for some of the abstract concepts used in ezCar2X.

Even though ns-3 provides several mobility models out of the box, none of them are specifically designed for vehicular networks. In addition, applications may require influencing the driver behavior based on received messages during the simulation. Our solution includes the open source traffic simulator SUMO [SUMO14] that can be coupled with other simulators using its well documented TraCI API [WPRH+08]. Both simulators are synchronized on a microsecond level with a mapping of nodes in ns-3 to vehicles in SUMO. The movement of the vehicles directly corresponds to changes in the mobility model of the assigned ns-3 nodes whereas applications can use TraCI to influence the driver model of the vehicle, e.g., force it to slow down due to a received warning about a dangerous situation ahead.

It should be noted that the integration of SUMO with ns-3 is independent from the combination of ns-3 and ezCar2X. Both tool sets can be used separately or in combination. In our approach, the integration of ezCar2X only concerns ns-3. The interaction with SUMO is provided completely through abstractions already available in ns-3, e.g. the node mobility model.

4.2 Integration of ezCar2X into ns-3

In general ezCar2X components can be used out of the box in combination with ns-3 since both tools are written in C++ and consist of a set of shared libraries. Simulations in ns-3 are usually written in C++ as well, thus, integrating ezCar2X means including the right header files in your

simulation script and linking against the required framework libraries. However, to use the full potential of ns-3 our integrated solution has to follow some of its architectural concepts. In addition, ns-3-specific implementations of a few basic ezCar2X concepts are required to access information and functionality available in the simulation environment, e.g., position of a node or the current time.

In ns-3 each simulated station is called a node. Every node has a selection of network devices and several other models attached to it. In our approach, a node will have at least one wireless interface, a mobility model and its own instance of the ezCar2X framework including various components configured by the simulation script. Furthermore, each node has a TraCI adapter that is used to map the node to a specific vehicle in the attached SUMO instance. Fig. 2 illustrates the basic setup.

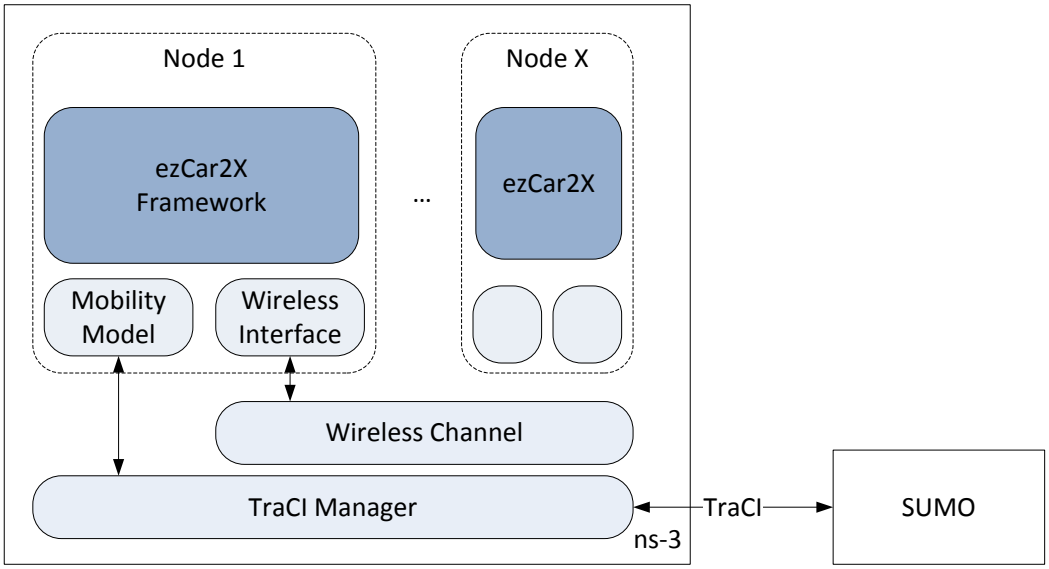

Fig. 2: Integration of ezCar2X, ns-3 and SUMO

Most components in ezCar2X require a basic set of functions. This does not only include access to time and position information, but also a communication stack to transmit and receive data packets. Furthermore, components in ezCar2X are usually event-driven, either by input received from external sources or by asynchronous events and timeouts provided by an event scheduler. The former are functions or data provided by ns-3 through one of its models, while the latter is already very similar to the discrete event scheduling used in ns-3. A complete integration of

ezCar2X requires ns-3-specific implementations of the basic concepts used in ezCar2X to access this functionality.

The position of a node is available through the ns-3 mobility model. A simple wrapper implementing the ezCar2X *PositionProvider* interface enables direct access to the current location of a station. A coordinate transformation has to be applied because node positions in ns-3 are provided as cartesian coordinates in three-dimensional space but position information in ITS applications is usually based on the World Geodetic System (WGS84).

The adaption of the *TimeProvider* interface is quite similar. However, the current simulation time can directly be accessed using the global *Simulator* instance in ns-3.

The wireless interfaces of a node in ns-3 can be made available to the ezCar2X components with an implementation of the *ItsG5Interface* that forwards send requests to its ns-3 pendant and invokes its *OnPacket* signal for each packet received from ns-3.

External events are already governed by ns-3's discrete event simulation since packet transmission is realized through the models provided by ns-3. Internal events, however, need to be handled by the ns-3 event scheduler as well. Therefore, we provide an implementation of the *EventScheduler* interface of ezCar2X that does not schedule events and invoke their handlers on its own, but rather forwards them to the ns-3 event scheduler and takes care of the callback conversion between ns-3 and ezCar2X whenever an event is due.

The described components are the basis for most components in ezCar2X and suffice for a wide range of applications. Scenario setup and result evaluation is further improved with a set of tools that automate the different steps required to install ezCar2X on many virtual nodes at once. These tools also take care of the assignment of values and data unique for each individual node, like address or path to the stored security certificates. The evaluation of the simulation results is assisted by several means to collect statistical data from different sources on all layers during a simulation run.

5 Conclusion and Outlook

We presented our integrated approach for simulation and prototyping of applications based on Car2X communication: the software framework ezCar2X. Provided as a set of libraries it covers many aspects of cooperative ITS applications. We have shown that due to its modular and hardware independent approach ezCar2X components can be used in a simulation environment

as well as on real world hardware without changes to the code base. The combination of ezCar2X with the state-of-the-art open source tools ns-3 and SUMO provides a comprehensive tool for extensive simulation campaigns. Thus, novel applications and protocols can be evaluated with fewer errors, quicker implementation and more reliable results obtained from simulation and real world deployment. Therefore, the ezCar2X framework is a sound basis for the development of applications in the fields of advanced driver assistance systems, cooperative transport systems and e-mobility. It can also provide valuable insights for the design of next generation ITS communication protocols.

With the research field greatly in motion there are many aspects not yet covered by ezCar2X. Various improvements of the framework are planned for the near future. These include integration of unified geocasting services over heterogeneous access technologies, advanced local dynamic map (LDM) facilities as well as full IPv6-over-GeoNetworking support. In addition, we intend to compare our simulation approach with the results obtained by existing simulation tools using real world examples. Finally, we plan an investigation regarding scalability of the proposed combination of ns-3, SUMO and ezCar2X for large scale simulations.

Literature

[BR14] Bittl, S. and Roscher, K.: Towards methodologies for rapid-prototyping of communication technologies and cooperative ITS applications to be used both on real target and inside simulation environments, as realized by a combination of the ezCar2X framework with the network simulator ns-3. In: The Third International Conference on Advances in Vehicular Systems, Technologies and Applications VEHICULAR, June 2014.

[CarM14] IPG: CarMaker, http://ipg.de/simulationsolutions/carmaker/, 2014.

[C2C12] CAR 2 CAR Communication Consortium: European vehicle manufacturers working hand in hand on deployment of cooperative Intelligent Transport Systems and Services (C-ITS). Press release, October 2012.

[ETSI10] Intelligent Transport Systems (ITS); Communications Architecture, ETSI EN 302 665 V1.1.1, 2010.

[ETSI12] Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service, ETSI EN 302 637-3 Draft, 2012

[ETSI13a] Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service, ETSI EN 302 637-2 Draft, 2013

[ETSI13b] ETSI ITS Cooperative Mobility Services Event, http://www.etsi.org/news-events/past-events/665-plugtests-2013-itscms3, 2013.

[Fest09] Festag, A. et al.: CAR-2-X Communication for Safety and Infotainment in Europe. In: NEC Technical Journal, vol. 3, no. 1 , 2008.

[Gray09] Gray, P. et al.: DSRC Field Trials. Whitepaper, Cohda Wireless, Aug. 2009.

[iTet14] iTetris, http://www.ict-itetris.eu, 2014.

[JiST14] JiST/SWANS, http://jist.ece.cornell.edu, 2014

[Laso08] Lasowski, R. et al.: OpenWAVE Engine / WSU - A Platform For C2C-CC: In: 15th World Congress on Intelligent Transport Systems, New York, Nov. 2008

[Ns214] ns-2, http://www.isi.edu/nsnam/ns, 2014.

[Ns314] ns-3, http://www.nsnam.org, 2014.

[OMNe14] OMNeT++, http://www.omnetpp.org, 2014.

[OSM14] OpenStreetMap, http://www.openstreetmap.org, 2014.

[PreS14] PreScan, https://www.tassinternational.com/prescan, 2014.

[Rive14] Riverbed Modeler, http://www.riverbed.com/products/performance-management-control/network-performance-management/network-simulation.html, 2014

[RJGH13] Roscher, K., Jiru, J., Gonzalez, A.A., and Heidrich, W.A.: ezCar2X: a modular software framework for rapid prototyping of C2X applications. In: 9th ITS European Congress, Dublin, Ireland, 4-7 June 2013.

[SRJ13] Steiner, T., Roscher, K., and Jiru, J.: Cooperative Glare Reduction using V2X radio technology. In: Int. Symp. Wireless Vehicular Communication WiVeC2013

[SUMO14] SUMO - Simulation of Urban MObility, http://sumo-sim.org, 2014.

[Vein14] Veins - The open source vehicular network simulation framework, http://veins.car2x.org, 2014.

[VISS14] PTV Vissim, http://vision-traffic.ptvgroup.com/en-us/products/ptv-vissim, 2014.

[VSim14] VSimRTI - Smart Mobility Simulation, https://www.dcaiti.tu-berlin.de/research/simulation, 2014.

[WPRH+08] Wegener, A., Piorkowski, M., Raya, M., Hellbrück, H., Fischer, S. and Hubaux, J.P.: TraCI: A Framework for Coupling Road Traffic and Network Simulators. In: Proceedings of the 11th Communications and Networking Simulation Symposium, April 2008.