Conference Paper in the Proceedings of 2008 IEEE Asia-Pacific Services Computing Conference (VON 2008)
15 Sep 2008, Los Alamitos, CA, USA

Realistic Simulation of V2X Communication Scenarios

Tobias Queck*†, Björn Schünemann*†, Ilja Radusch† Prof. Christoph Meinel*
Email: {tobias.queck, meinel}@hpi.uni-potsdam.de, {bjoern.schuenemann, ilja.radusch}@dcaiti.com

*University of Potsdam
Hasso-Plattner Institute
Prof.-Dr.-Helmert-Straße 2-3
14482 Potsdam, Germany

†Technische Universität Berlin
OKS / Daimler Center for Automotive IT Innovations
Ernst-Reuter-Platz 7
10587 Berlin, Germany

***Abstract*—Vehicle-2-X (V2X) Communication provides the foundation for new applications that enhance both safety and traffic efficiency. Before V2X applications can be deployed in practice, their in-depth analysis is necessary. For this end, detailed and realistic simulations are essential. Depending on the simulated V2X Communication application, particular simulators have to be coupled. For this purpose, we have developed the V2X Simulation Runtime Infrastructure (VSimRTI) offering the flexibility to combine arbitrary simulators. The VSimRTI is derived from concepts of the High Level Architecture (HLA). It synchronizes the simulators and enables the communication among them. Another feature of our simulation environment is the emulation of the environment of V2X Communication applications in real vehicles. As a result, we can integrate real V2X Communication applications without modifications.**

I. Introduction

Currently, one aim of automotive research has been to establish vehicular communication, based on wireless short-ranged networks, to enhance both safety and traffic efficiency. Therefore, various V2X Communication applications are deployed in vehicles. To evaluate the improvements that can be achieved by these applications, scenarios are defined and field tests are carried out. However, the realization of field tests is rather complex and expensive. So, simulations are essential to prepare the tests in the real world and reduce their costs.

In [1], Schünemann et al. show that different simulators have to be combined to provide a simulation environment for a more realistic simulation of V2X Communication scenarios. Based on this requirement, one challenge is to run simulators synchronously. The clocks of the simulators have to be synchronized and, moreover, simulation data has to be exchanged among simulators. Several existing simulator couplings use particular simulators and do not offer the opportunity to exchange them. For the simulation of different V2X Communication applications, this is not satisfying since requirements vary depending on the different applications.

Another challenge is the realistic execution of V2X applications in the simulation environment. For an attempt at a solution, two different approaches exist. First, only the behaviour of the application can be simulated without the integration of the programming code of this application. Second, the real application can be executed by integrating the programming code and emulating the interfaces of the environment of the application. In this case, there is no difference from the point of view of the application if the application runs in a simulation environment instead of in a real vehicle. Hence, the second approach is more realistic for the simulation of V2X applications than the first one.

To master the challenges described above, we developed the V2X Simulation Runtime Infrastructure (VSimRTI). Our simulation infrastructure allows the integration of time-discrete simulators, e.g. for network, traffic, and environment simulation. It couples the simulators and provides the flexibility to exchange them depending on the specific requirements of a simulation. For the synchronization of and communication among all components, the VSimRTI uses concepts of the IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) [2]. Moreover, our simulation environment contains an application container that emulates the environment of V2X applications in real vehicles. Consequently, real V2X applications can be integrated without adapting them for our simulation infrastructure.

A. Related Work

Currently, a number of projects has already combined different kinds of simulators for the simulation of V2X Communication scenarios. An Open Source implementation that integrates the traffic simulator SUMO [3] and the network simulator ns-2 is TraNS [4]. It enables the direct interaction of both simulators. A further project is the Multiple Simulator Interlinking Environment for C2CC in VANETs (MSIECV) [5] that integrates the network simulator ns-2, the traffic simulator VISSIM, and Matlab/Simulink for application level simulation. In both projects, it is possible to influence the vehicle behaviour, e.g. to stop or reroute vehicles.

A different approach to creating a simulation environment is to extend a single simulator with all missing functionality. The VANET Simulation Environment [6] is an integrated simulation environment on top of the network simulator JiST/SWANS [7]. The original random way point model in SWANS is replaced by a more accurate traffic model.

All these projects do not offer the user the possibility to exchange simulation components by other suitable simulators.

They allow the simulation of application behaviour, but do not provide the opportunity to integrate applications developed for real vehicles.

B. Structure of this Paper

This paper is organised as follows: Section II illustrates the concepts on which our architecture is based. In Section III, we introduce the components of our VSimRTI. Thereafter, in Section IV, we explain how a real V2X Communication application is evaluated in the VSimRTI. Finally, in Section V, we describe the simulation of a V2V decentralized environmental notification application and present the simulation results.

II. Simulation Concept

The simulation of V2X Communication scenarios involves various aspects. In particular, a microscopic traffic simulator is used to simulate the movements of the vehicles. Moreover, a network simulator simulates the wireless communication among the vehicles and an application simulator provides the environment for simulating a V2X application. Currently, several accepted solutions exist to simulate either vehicle traffic, wireless networks, or the behaviour of an application. But, a key requirement for a realistic V2X Communication simulation is the coupling of all these simulators and the interaction among them at runtime of the simulation [1]. In the following subsections, we motivate our concepts to fulfil this requirement.

A. Flexibility

To run simulations as realistically as possible, it is necessary to combine simulators of arbitrary domains with each other. So, particular simulators are necessary for the simulation of particular applications. For example, simulators for traffic, network, application, and environment simulation are required for a V2X application responding to weather conditions. In contrast, instead of an environment simulator, a traffic light scheduler is necessary for the simulation of a green-wave scenario. Another aspect is that simulators of the same domain offer different features for specific scenarios, e.g. traffic simulators especially developed for either highway scenarios or inner-city traffic exist.

B. High Level Architecture (HLA)

A standardized approach to combine simulators with each other is the IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) [2]. It consists of rules, a runtime infrastructure, interface descriptions, and an object model template. In this standard, the coupling of different simulators is a federation that satisfies the defined rules. These rules consist of requirements for participating simulators and for the overall federation which regulate the object management and the interaction among simulators. The central component of each federation is a runtime infrastructure (RTI). It connects all participating simulators that are encapsulated by federates and controls the simulation run. The RTI and each federate implement ambassadors to handle the communication between them. The runtime infrastructure provides a set of services available to the federates. The services manage the coordination of federation-wide activities; the synchronization of federates; and the generation, modification, and distribution of information, objects, and interactions. The Object Model Template (OMT) defines the way of documenting federations and federates. The HLA object models are formal definitions of the data transferred among federates.

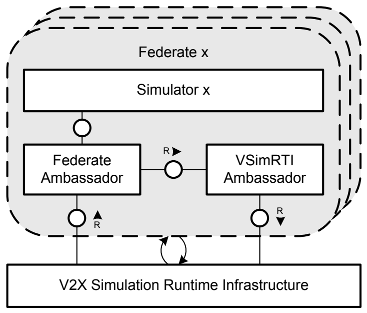

Fig. 1. Communication among VSimRTI and Simulators

C. V2X Simulation RTI

Inspired by the HLA standard, we have developed the V2X Simulation Runtime Infrastructure (VSimRTI) for the simulation of V2X Communication scenarios. We have implemented selected parts of the HLA standard and have used its concepts for our simulation runtime infrastructure. The VSimRTI offers services which handle synchronization and communication of the federates, the management of global data, and the lifecycle management of federates. The communication among the federates is enabled by the VSimRTI which is accessible by ambassadors similar to the HLA standard. The types of data to be exchanged are defined as an extensible set of messages instead of using the general OMT of HLA. [8]

III. VSimRTI Architecture

To run a simulation, a federation of simulators has to be created. This federation consists of one federate for each participating simulator. In the upper part of Figure 1, the inner structure of a federate is illustrated. It consists of the original simulator that is connected to its federate ambassador and to an instance of a VSimRTI ambassador. The federates run on top of the V2X Simulation Runtime Infrastructure (see lower part of Figure 1) which offers federation, interaction, time, and vehicle management. The complete communication among federates is handled by their ambassadors using messages. Each federate provides an ambassador offering an interface that is used by the VSimRTI to control the simulator and to provide messages from other federates.

A. Federation Management

The federation management is responsible for the administration of participating federates. This includes deploying, starting, stopping, and undeploying federates in a distributed system. Before running a simulation, the federation management creates an empty federation. After that, federates join the federation. Already joined federates can be removed from a federation, if they are no longer necessary for the simulation. After a simulation is finished, the federation management frees used resources.

B. Time Management

The main problem during the execution of a federation is the synchronization of its federates. Each federate runs a discrete event simulation where, sequentially, the first event of an ordered event list is processed. Processing an event can include sending messages to other federates using the interaction management that results in new events in other federates. Consequently, a time management is necessary for coordinating the simulation and synchronizing participating federates. It ensures that each federate processes its events in the correct order. In [8], we describe the time management of the VSimRTI in more detail.

C. Vehicle Management

The vehicle management is responsible for the storage and synchronization of vehicle data relevant in several federates. We decided to store this data inside the VSimRTI because it is used in most of the V2X related simulators. To avoid inconsistent data, only one federate is allowed write access at a time. Any federate is allowed to read data when no federate has approved write access. If there are concurrent read or write access claims, the federates are scheduled in a first-come-first-served manner.

D. Interaction Management

The exchange of data among federates is provided by the interaction management using messages. The VSimRTI and its federates are decoupled through a publish-subscribe pattern provided by the interaction management. A published message is forwarded to each subscribed federate directly after it has been published. A message consists of its creation time, an identifier describing its type, and optional data. Messages to exchange traffic, network, vehicle, and sensor data are predefined. These messages are used to define a standardized communication behaviour.

IV. Application Integration

Independent of the simulation runtime infrastructure and the integrated simulators, the core of a V2X Communication scenario consists of the vehicles and their running applications. For the simulation of V2X Communication scenarios, we categorized vehicles according to their ability to participate in the communication. We will describe this categorization in detail in the following subsection. Based on their category, several simulators are necessary to simulate all properties of a vehicle.

The main objective of our work is to evaluate applications that are to be embedded in future vehicles. Consequently, real V2X Communication applications have to be integrated in the simulated scenario without modification. Accordingly, every interface of a component that is available for an application in a vehicle must also be available during a simulation run. Hence, VSimRTI provides interfaces that encapsulate all in- and outgoing data of an application so that this requirement is fulfilled.

A. Vehicle Categories

We classify vehicles into three different categories. The first category is the classic vehicle which covers vehicles without hardware components for vehicular communication. Accordingly, these vehicles cannot communicate with their neighbourhood. In contrast to classic vehicles, equipped vehicles contain a wireless network device and lower communication layers to participate in V2X Communication networks. These vehicles are able to send, route, and receive V2X Communication messages. Yet, equipped vehicles have no applications deployed to handle the received data. The third category, application-supported vehicles, have an application layer on top of the lower network layers. This layer allows to run applications that can communicate with other vehicles using the communication stack in their host vehicle.

B. Simulation of Vehicles and their Components

For the simulation of a classic vehicle, both its characteristics and its route are specified. Therefore, position, direction, and speed of a vehicle have to be simulated at every specific time of the simulation process. Consequently, a default traffic simulator is adequate to simulate classic vehicles. In contrast, an equipped vehicle needs an additional simulation of its network stack. As a consequence, a network simulator is necessary to simulate the wireless communication. If V2X Communication applications are to run on a vehicle, the complexity of the simulation increases significantly. All potential interactions among applications and the hardware of the vehicle have to be emulated. To integrate applications implemented for real vehicles, an application interface simulator is necessary which offers exactly the same interfaces that the runtime environment of the vehicle provides.

C. Simulation Environment for Applications

In general, an application deployed in a vehicle has no direct access to vehicle specific components. An application container or a framework is used creating an abstraction of common V2X components. As a consequence, we have modified the corresponding application container to integrate it into our V2X Simulation Runtime Infrastructure. So, the interfaces for the communication among applications and underlying components are implemented by an adapter connected to our runtime infrastructure (see Figure 2).

D. Application Model

To integrate applications into a time-discrete simulation, an abstract model of V2X Communication applications is necessary. Applications can have two possible states at runtime: idle or active. An idle application becomes active either driven by an external event (incoming message, environment change, or vehicle data) or by fulfilling a recurring task. While an

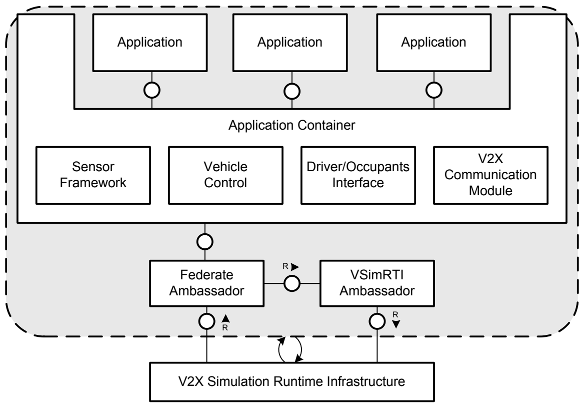

Fig. 2. Integration of V2X applications in VSimRTI.

application is active, it requests environment or vehicle data, or it sends V2X messages to other vehicles or infrastructure components. After an active period, an application becomes idle again. The underlying application container is responsible for switching between the states. Incoming messages, generated by simulators, are converted into events that trigger an application state change. Additionally, a timer event is needed to schedule recurring active applications. When an active application sends a message or requesting additional sensor data, its container generates a message to forward the data to a corresponding simulator. To synchronize an application container with other federates, the events are scheduled at the time management service.

V. Evaluation of example Application

Before V2X applications can be deployed in practice, an in-depth analysis is necessary. For example, the assumed reduced driving time that can be achieved by a V2X application is one important factor that has to be investigated. Moreover, the minimum necessary coverage of application-supported vehicles in order to have a measurable benefit is meaningful. To show the usability of our simulation infrastructure for the evaluation of V2X applications, we implemented a V2V decentralized environmental notification application [9]. This application broadcasts warnings if a slippery road has been detected by the vehicle. Other vehicles, receiving this warning message, try to optimize their routes by circumnavigating the dangerous location. The application runs in an application container which is registered as a federate (see Figure 2) besides a traffic simulator, a network simulator, and an environment simulator.

A. Simulators

We use the open source simulator SUMO to simulate the vehicle traffic. SUMO offers the runtime interface TraCI [10] to control and influence the vehicle behaviour at runtime of the simulation. Hence, the corresponding ambassador converts messages and management commands into a TraCI compatible format. Moreover, the traffic generated by SUMO is used as input for the network simulator JiST/SWANS. We have implemented an extension for JiST/SWANS that allows us to synchronize the internal scheduler, modify node positions, and send and receive packages using a socket interface. The corresponding ambassador connects these extensions with our simulation infrastructure. The third integrated component is the environment simulator eWorld[1]. We use eWorld to import real road map data from OpenStreetMap[2]; enhance them with location-based information, like slippery roads; and export them as SUMO input files. In addition, we have implemented an ambassador which is connected to the eWorld EventServer to request location-based information at runtime of a simulation to emulate the sensors of a vehicle. This concept is used, in particular, to emulate the detection of e.g. a slippery road by the responsible sensor of a vehicle. The fourth component is a simple in-house application container. This container is a Java implementation based on requirements defined in the C2C-CC Manifesto [9]. We have implemented an ambassador for that and run the modified container with JiST. All connections to hardware components are replaced with chronologically ordered calls to the corresponding ambassador. This ambassador forwards the calls as messages to subscribed simulators. Additionally, we have implemented a component allowing a user to define the ratio of classic vehicles, equipped vehicles, and application-supported vehicles. During runtime, this component assigns the category of each newly created vehicle. For application-supported vehicles, an additional application container is started and its corresponding ambassador is registered at the VSimRTI.

B. Simulated Scenario

Our use case aims at avoiding dangerous situations and reducing traffic congestion with the help of a V2X Communication application. For our simulation, we use real road map data of an area around the German motorway interchange *Autobahndreieck Schwanebeck* to the north-east of Berlin. In our scenario, traffic stagnates because of a slippery road segment on the motorway. Since, a slippery road can result in accidents, our implemented V2V decentralized environmental notification application makes sure that vehicles broadcast warnings when they detect such a slippery road. As a result, the following vehicles reduce their speed. To avoid congestion, vehicles still at a greater distance try to reroute and circumnavigate the dangerous location by leaving the motorway once they receive the warning.

Before the simulation run, the ratio of the different vehicle categories is defined. For example, a simulation can have 30% of classic, 40% of equipped, and 30% of application-supported vehicles. During the simulation, the driving time of each individual vehicle is measured. After the simulation, the average driving times of all vehicles, depending on their categories, are calculated.

C. Results

In this subsection, we present our simulation results for detecting the minimum necessary coverage of equipped and

[1]http://eworld.sourceforge.net
[2]http://www.openstreetmap.org/

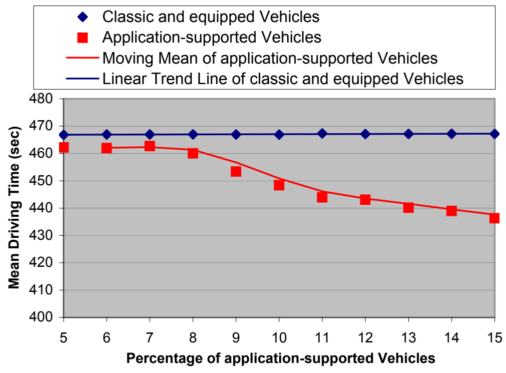

Fig. 3. Influence of ratio of application-supported vehicles on driving time

application-supported vehicles in order to decrease the driving time of the application-supported vehicles measurably.

In Figure 3, the relation between average vehicle driving times and percental coverage of application-supported vehicles is illustrated. The red line shows the driving times of application-supported vehicles and the blue line illustrates those of the others. In these simulation runs, all vehicles were equipped with V2X hardware so that the results are not influenced by an insufficient number of equipped vehicles. The results show a marginal driving time decrease of less than 5 seconds if the coverage of application-supported vehicles is between 5% and 7%. From 8%, the driving time starts to decrease continuously. With 9% coverage, the average driving time of application-supported vehicles is decreased by 13 seconds.

Classic (%)	Equipped (%)	Application-supported (%)	Driving Time (sec)
0	91	9	453.39
...	...	...	...
71	20	9	453.71
81	10	9	454.07
91	0	9	453.62

TABLE I
VARYING THE COVERAGE OF EQUIPPED VEHICLES.

In the following simulation runs, we evaluated the influence of the ratio between classic and equipped vehicles (cf. Table I). Here, we used 9% of application supported vehicles. Our results show that the driving times of application-supported vehicles change insignificantly when varying the ratio between classic and equipped vehicles. Hence, no improvement can be achieved if additional equipped vehicles are present that can forward V2X communication messages, but have not deployed the application.

VI. Conclusion

In this paper, we presented our V2X Simulation Runtime Infrastructure VSimRTI. With the VSimRTI, we have created an infrastructure to couple arbitrary discrete event-based simulators. We solved the synchronisation and interaction problem of coupled simulators by using concepts of the HLA standard. The VSimRTI provides the flexibility to exchange simulators without changing the infrastructure. Furthermore, our infrastructure able us to evaluate applications that will be deployed in future vehicles, i.e. a modification of the real applications for our simulation architecture is not necessary. Moreover, we simulated a V2X Communication scenario where the influence of a V2V decentralized environmental notification application was evaluated to increase the traffic flow. In this scenario, we used the traffic simulator SUMO, the network simulator JiST/SWANS, the environment simulator eWorld, and a modified in-house application container to integrate the application.

A. *Future Work*

Currently, we plan to simulate further scenarios, e.g. traffic flow improvements that can be achieved by a green-light advisory application. Moreover, it is planned to optimize the synchronization of simulators and to integrate further simulators into our infrastructure. Accordingly, we plan to integrate the network simulator OMNeT++ and the traffic simulator VISSIM. In addition, we aim to offer an optimistic synchronisation mechanism. Therefore, federates have to provide functions to snapshot and roll-back their internal states.

References

[1] B. Schünemann, K. Massow, and I. Radusch, "A novel approach for realistic emulation of vehicle-2-x communication applications," *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, pp. 2709–2713, May 2008.

[2] Institute of Electrical and Electronics Engineers, *IEEE standard for modeling and simulation (M&S) High Level Architecture (HLA)–federate interface specification. IEEE Standard 1516.1.* New York, NY, USA: IEEE, 2000.

[3] D. Krajzewicz, M. Bonert, and P. Wagner, "The open source traffic simulation package sumo," 2006.

[4] M. Piorkowski, M. Raya, A. Lugo, P. Papadimitratos, M. Grossglauser, and J.-P. Hubaux, "TraNS: Realistic Joint Traffic and Network Simulator for VANETs," *ACM SIGMOBILE Mobile Computing and Communications Review*.

[5] C. Lochert, M. Caliskan, B. Scheuermann, A. Barthels, A. Cervantes, and M. Mauve, "Multiple simulator interlinking environment for inter vehicle communication," in *VANET 2005: Proceedings of the Second ACM International Workshop on Vehicular Ad Hoc Networks, Cologne, Germany*, September 2005, pp. 87–88.

[6] C. Gorgorin, V. Gradinescu, R. Diaconescu, V. Cristea, and L. Iftode, "An integrated vehicular and networking simulator for vehicular adhoc networks," in *20th European Simulation and Modelling Conference, Toulouse*, 2006.

[7] R. Barr, Z. J. Haas, and R. van Renesse, "Jist: an efficient approach to simulation using virtual machines: Research articles," *Softw. Pract. Exper.*, vol. 35, no. 6, pp. 539–576, 2005.

[8] T. Queck, B. Schünemann, and I. Radusch, "Runtime infrastructure for simulating vehicle-2-x communication scenarios," in *VANET '08: Proceedings of the Fifth ACM International Workshop on Vehicular Ad Hoc Networks*. New York, NY, USA: ACM, September 2008, to appear.

[9] R. Baldessari, B. Bödekker, M. Deegener, A. Festag, W. Franz, C. C. Kellum, T. Kosch, A. Kovacs, M. Lenardi, C. Menig, T. Peichl, M. Röckl, D. Seeberger, M. Straberger, H. Stratil, H.-J. Vögel, B. Weyl, and W. Zhang, "Car-2-car communication consortium - manifesto," p. 93, 05 2007.

[10] A. Wegener, M. Piorkowski, M. Raya, H. Hellbrück, S. Fischer, and J.-P. Hubaux, "TraCI: An Interface for Coupling Road Traffic and Network Simulators," in *11th Communications and Networking Simulation Symposium (CNS'08)*, 2008.