VIRTUAL COMMUNICATION STACK: TOWARDS BUILDING INTEGRATED SIMULATOR OF MOBILE AD-HOC NETWORK-BASED INFRASTRUCTURE FOR DISASTER RESPONSE SCENARIOS

Aznam Yacoub

Polytechnique Montreal, Heterogeneous Embedded System Laboratory, Montreal, QC, Canada
Aix Marseille Université, Université de Toulon, CNRS, LIS, Marseille, France
aznam.yacoub@polymtl.ca

ABSTRACT

Responses to disastrous events are a challenging problem, because of possible damages on communication infrastructures. For instance, after a natural disaster, infrastructures might be entirely destroyed. Different network paradigms were proposed in the literature in order to deploy adhoc network, and allow dealing with the lack of communications. However, all these solutions focus only on the performance of the network itself, without taking into account the specificities and heterogeneity of the components which use it. This comes from the difficulty to integrate models with different levels of abstraction. Consequently, verification and validation of adhoc protocols cannot guarantee that the different systems will work as expected in operational conditions. However, the DEVS theory provides some mechanisms to allow integration of models with different natures. This paper proposes an integrated simulation architecture based on DEVS which improves the accuracy of ad hoc infrastructure simulators in the case of disaster response scenarios.

Keywords: DEVS, Simulation Tools, Mobile Ad Hoc Networks, Verification and Validation, Disaster Response.

1 INTRODUCTION

Mobile Ad Hoc Networks (MANETs) have been proposed in the litterature (Kiess and Mauve 2007, Reina et al. 2015, Mohammed and Al-Ghrairi 2019) as a communication technology in the case of emergency and disasters. Indeed, cellular-based infrastructures might become unavailable due to important damages. While MANETs can be quickly deployed without fixed infrastructure, setup or prior requirements, their flexibility is attractive when communications between victims and rescue teams are crucial. However, their implementations face an important challenge: proving that they are enough reliable compared to other approaches (Kiess and Mauve 2007). While Verification and Validation (V&V) using real experimentations in emergency conditions is utterly impossible, simulation is an important tool in the MANET research community.

Simulators are an inexpensive manner to evaluate the performance and the accuracy of algorithms and systems without the use of the actual hardware. Also, simulators allow checking the capacity of a network in extreme conditions by varying various parameters in a virtual way and checking different scenarios (Manpreet and Malhotra 2014). However, although their use and development increased, the credibility of their results decreased over the time (Kurkowski et al. 2005, Hogie et al. 2006). Among the problems encountered

<Conference Acronym>, <Month DD-DD>, <Year>, <City, State, Country>;
©*<Year> Society for Modeling & Simulation International (SCS)*

during the development of MANETs, some are inherent to simulation in general: repeatability, consistency, and accuracy of the models (Sargent 2001). Particularly, simulators generally focus only on some aspects of the network structure itself without taking into account the complexity and the heterogeneity of the systems which rely on this network: autonomous vehicles, unmanned aircraft systems, communication software, etc.

For instance, Figure 1 shows an exemple of real MANET-based ecosystem in an emergency situation. Collaborative drones evolving in a complex environment must communicate without a fixed network infrastructure, send data to different rescue teams with real-time 3D processing software on mobile devices in order to allow professionals to evaluate the situation. Then, these data should also be saved in a database connected to internet in order to allow management teams to take important decisions. Decision support can also be assessed thanks to an Artificial Intelligence-Driven Decision Making Process (Phillips-Wren and Jain 2006). In other words, the verification and the simulation of the entire ecosystem should take into account all the different aspects and natures of all the devices and disasters. This is obviously impossible, but *abstraction* is admitted as a real problem especially in the case of MANET simulation (Hogie et al. 2006). Moreover, complex and heterogeneous collaborative systems imply the use of various kinds of models. Therefore, some of these components can be modelled using discrete-event models, continuous models, automata, etc.

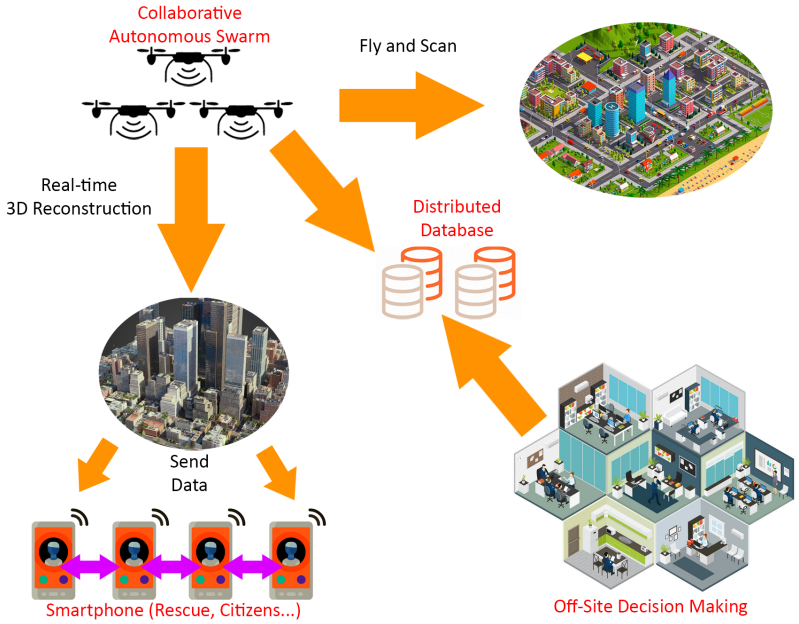


Figure 1: An example of heterogeneous ecosystem communicating using MANET.

Various areas addressed the problem of making heterogenenous simulators coexisting and working together in order to improve the accuracy of simulations, and to deal with repeatability and consistency. Essentially techniques like cosimulation (Vaubourg et al. 2015, Gomes et al. 2018) have been proved as good approaches that allow modelers to take into account specificities of different subsystems. However, problems related to repeatability, accuracy and scalability must still be resolved in the case of MANET simulators as stated in the previous cited articles. Especially, the Theory of Modelling and Simulation (TMS), and in particular the Discrete-Event System Specifications (DEVS) formalism (Zeigler et al. 2000, Zeigler et al. 2019), provides foundations which allow the building of heterogeneous simulators in a hierarchical manner. This kind of simulators allows embedding models with different levels of abstraction, if they respect the DEVS principles. If all the surveys stated in this article show that MANET simulators generally use a discrete-event paradigm, there were few attempts to apply the TMS and DEVS in the case of MANET modelling and simulation.

In this paper, we propose general guidelines and insights to improve the simulation of MANETs-based infrastructure by using DEVS approaches and results. We show that some results in the DEVS area can benefit to MANET simulation by making easier cosimulations and by making different levels of abstraction coexist inside a unique integrated environment. The first section recalls the existing work concerning MANET simulators and DEVS architecture. In the second section, we introduce our proposed integrated simulation architecture which allows switching between Software-in-the-Loop (SIL) and Hardware-in-the-Loop (HIL) paradigms (Murray-Smith 2012) by using the DEVS Bus concept (Yong Jae Kim and Tag Gon Kim 1998, Kim et al. 2003). Implementation details and detailed examples are outside the scope of this paper, and are developed in another one.

2 RELATED WORKS

Literature about MANET simulation can be splitted in two essentials parts. The first one concerns the common used architectures in the case of network simulation. The second one concerns the actual techniques known in the area of Modelling and Simulation (M&S), DEVS and simulation theory.

2.1 MANETs Simulators Architectures

MANETs simulation can be essentially overviewed by looking at the surveys (Manpreet and Malhotra 2014, Dorathy and Chandrasekaran 2019) which show that simulator architectures have intensively been studied (Kurkowski et al. 2005, Mallapur and Patil 2012, Chengetanai and O'Reilly 2015) without having really evolved for decades. Indeed, as stated by Kurkowski et al. (2005), Andel and Yasinsac (2006), this kind of network represents a challenge for simulation community. The complexity is mainly induced by two specific aspects (Gunes et al. 2007):

- The first one concerns the frequence of the topological modifications. Topology of the network in the case of MANET is related to mobility. However, mobility models generally rely on unrealistic assumptions (randomizations, Manhattan model, etc.). While mobility was understood as having a non-negligible impact on the accuracy of simulations (Schindelhauer et al. 2003), mobility models have been specifically studied in separate works (Sichitiu 2009, Khairnar and Pradhan 2011).
- The second one concerns the modelling of physical phenomena. Indeed, MANETs generally rely on wireless communication, which implies radio propagation modelling. As stated in (Hogie et al. 2006), study of waves propagation is a complex problem which needs elaborated techniques (Schmitz and Wenig 2006). These techniques make simulations run slower. Therefore, a lot of MANET simulators make strong assumptions and provide simplified models for physical interactions (Andel and Yasinsac 2006).

Testbeds (Muchtar et al. 2018) can help researchers to overcome these two problems by making real experiments and consider these two models as controlled parameters. However, testbeds remain not scalable and cost expensive.

Therefore, these two problems are generally abstracted and the tradeoff between reduced accuracy and exe-cution speed is considered as acceptable. MANET researchers therefore develop simulation models which focus on two specific aspects: network performance tests and routing protocols comparison (Andel and Yasinsac 2006). In the cited article, the authors point that this approach makes wireless models suffer essentially of a lack of accuracy and inconsistencies with extreme divergences between simulators (Cavin et al. 2002). If we go further and beyond the existing analysis, we can easily understand the main reasons.

First, Andel and Yasinsac (2006) point out the use of unrealistic application traffic during simulation. Indeed, whereas Hogie et al. (2006) states that

> 'Software layers are relatively easy to re-implement within simulators ',

most of existing MANET simulators commonly use a three-layers protocol stack model (Figure 2) (Tüncel et al. 2016) which includes:

- A physical layer combining the physical layer and the data link layer;
- A network layer;
- An application layer combining all the layers above the network layer.

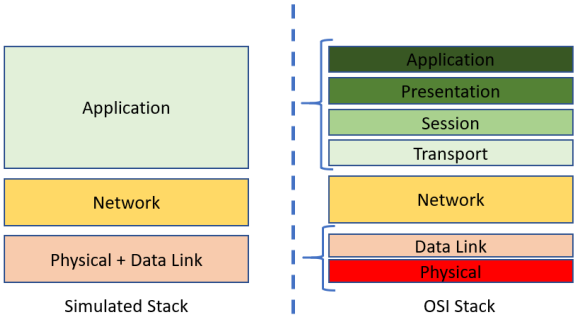


Figure 2: Simulation Stack (left) vs Real Stack (right).

This architecture could be a good abstraction if the cases of study were representative of the complexity of a software. However, Andel and Yasinsac (2006) stated that in most of studies, a constant-bitrate traffic generator is used in the layer 3 of the simulated stack while real software generally more depend on several complex interactions between internal and external components. Especially, traffic generation should take into account a wide range of parameters which depends on usage profile, application specificities, application performance, etc. This statement leads to questionning the assumptions made from the simulation using a *simplified* stack. If we don't question the abstraction of the application layer, we can really wonder what is the impact of this kind of simplification on the accuracy. Especially, in a fully connected environment with heterogeneous components, misuses of protocols can harmfully reduce the performance and can lead to question the robustness of a network. By pushing the reasoning further, interactions between stacks are even more complex in a virtualized environment, in which several stacks can be combined. For instance, in the case of a virtualized operating system, there are at least two communication stacks in the environment. These stacks can be in conflict, resulting a drastically change of the speed in the communication. Therefore, simulating such an environment using one simplified stack could not guarantee the efficiency of the modelled network.

Second, as we stated before, physical layer is generally modelled as controlled variables or using ideal conditions that are not reflecting the actual implementations. Takai et al. (2001) show the effect of such inaccurate models on the simulation. The experiment consists on evaluating the impact of physical layer settings on Ad Hoc On-Demand Distance Vector (AODV) (Perkins and Royer 1999) and Dynamic Source Routing (DSR) (Johnson et al. 2001) protocols in simulated environments. These factors signficantly affected the results of the simulation. More important, some settings changed the relative ranking between the protocols. A small variation on the underlying models changed the results of the qualitative comparison analysis between the evaluated protocols. If the result is expected and understandable, it raises an important statement: the level

of details is important in the case of MANET simulation (Hogie et al. 2006) and abstraction or refinement can lead to erroneous outcomes. This opens the way to two fundamental questions: the first one is related to the interpretation of the simulation results, and the second one is related to the V&V of models.

Indeed, studies show that this last problem is especially important in the case of MANET simulations. Kurkowski et al. (2005) demonstrate that generally researchers and developers use MANET simulators without prior checking that the used models were validated. Furthermore, when simulation results don't seem to be reliable, they modify the models without new validation steps. As a result, Andel and Yasinsac (2006) remind that routing protocols don't properly work in real world in many cases, while they produced good results in simulation. The same observation can also be stated when it comes to verification, at least for Pseudo Random Number Generator (PRNG). If models must be validated against user specifications, algorithm implementations should be also verified against requirements (Sargent 2011).

Another question is the lack of definition of the good level of abstraction when a model is developed, what Hogie et al. (2006) names the *granularity*, but also from the fact that is utterly impossible to develop enough meaningful scenarios in real-world. Consequently, modellers have no good comparison basis in the case of MANET development, especially in the case of disaster response. Indeed, most of testbed experiments involves less than 50 nodes (Hogie et al. 2006), whereas real situations imply hundred or thousand of nodes. Moreover, test and simulation scenarios mainly depend on experts, and imply the question of the coverage, which is out of the scope of this article. V&V of Simulation Models (Sargent 2001) are also related to the paradigms and the natures of the different models. Indeed, simulations are often carried from model with different natures, sometimes using discrete-event models, discrete-time models, continuous models, without clearly using a specified formalism. Especially, discrete-event paradigm is well-used for modelling the computational aspects but accurate physical models need continuous approaches. Using the wrong paradigm without motivating it by a serious analysis can lead to increase an undesirable and not necessary heterogeneity of the models. While existing simulators have all a their own purposes (Mallapur and Patil 2012, Manpreet and Malhotra 2014) and whereas integrating simulators is a hard challenge, researchers generally focus on one simulator and try to implement models in the paradigm of this simulator (Hogie et al. 2006). The outcome of such a practice is the blur of the choice of the good level of abstraction. By extension, it also brings problems while the heterogeneity exists also at the level of implementation, because all of these simulators have their own programming language. In other words, practices in MANET simulation lead to increase the heterogeneity of models over the heterogeneity of the modelled systems, and by extension the number of erroneous outcomes.

Then, the main problems encountered by MANET simulation studies can be splitted in four categories:

- The lack of proper studies when developing the MANET simulators, which leads to inconsistencies; this is more related to methodologies which tend to stick to a simulator instead of trying to develop accurate models for a proper purpose;
- The lack of verification and validation in the simulation development process;
- The difficult of choosing the right level of abstraction, which leads to oversimplification or overdetailed implementations;
- The unrealistic cases of application.

MANET simulation community is fully aware of these four problems. If the last one cannot be entirely resolved by a proper development methodology, especially in the case of emergency response, there are clues and insight to resolve the three others. First, Hogie et al. (2006) explicitely show the attempts to develop simulators at the application level: the first one is DIANEmu (Klein 2003) which provides an environment for simulating applications communicating through a network. However, DIANEmu doesn't simulate the four first layers of the network stack. JANE (Frey et al. 2004) tries to combine the advantages

of simulators, emulators and testbeds by providing a software which is able to work in hybrid mode thanks to a simulation environment and an execution platform. Then, an application can easily swap between the simulated network and the real devices while the communication interface is the same from the point of view of the software. However, the simulation models are themselves defined at a high level of abstraction. Consequently, JANE is not well-suited for a complex heterogeneous environment. However, the results show that the weaknesses of simulation environment introduced in the previous paragraphs can be overcome by introducing emulation aspects. The existence of emulators like JEMu (Flynn et al. 2001) and MANE (Ivanic et al. 2009) shows also the importance of the real tests in the evaluation of MANETs.

A close look at the simulation paradigm used by the well-used MANET simulators (Dorathy and Chandrasekaran 2019) shows that almost of them implement a discrete approach, in order to reduce the intrinsic complexity of the MANET analytic models. More precisely, some of them like OMNET++ (Varga and Hornig 2008) and NS3 (Riley and Henderson 2010) use a discrete-event-based architecture without explicitely or fully following the DEVS formalism (Zeigler 1976). This is particularly interesting because, as we show in the next section, the TMS (Zeigler et al. 2019) provides some recommendations which can help to resolve the problems stated previously. Therefore, the next section answers a crucial question: can the DEVS methodology help in more accurate modelling and simulation of MANET and can we provide a methodology to integrate existing MANET simulators into a DEVS-compliant environment ?

2.2 DEVS Methodology for Modelling, Simulation, Verification and Validation

TMS (Zeigler et al. 2000, Zeigler et al. 2019) gives guidelines for formalizing, modelling and simulating systems in a hierarchical, uniform and universal way. Indeed, the methodology advocates to see any systems as a composition of small black-boxes which take input called observable events, and react according to them. Moreover, each subsystem can also autonomously changes its own state at a specific time t, and output a corresponding event. In addition to that, TMS provides a clear separation between conceptual models and computerized (called also simulation) models. More formally, the theory provides a well-defined mathematical specification formalism for structure and behaviour of dynamic systems. DEVS conceptual models are therefore expressed using a clear algebraic structure. Basically, a DEVS model is composed by:

- **a DEVS Atomic model** which is a the most basic unit block. It is a state-machine which describe the behaviour of the component according to received or emitted events;
- **a DEVS Coupled model** which is a composition of DEVS models. Intuitively, it describes the relations and interactions between components.

Aside of the algebraic structure of the conceptual model, TMS defines the DEVS Abstract Simulation Model. This model offers the operational interpretation of the DEVS mechanisms. To each DEVS atomic model correspond a DEVS simulator, and to each DEVS coupled model correspond a DEVS coordinator. The DEVS simulator is organized in a tree way, in which the top root coordinator corresponds to the entire model of the system, and each internal node corresponds to a coordinator of a subcomponent. Leafs are the automata simulators which describe the behaviours of the system and subsystems. This architecture allows hierarchical description of models which makes easier the analysis. Furthermore, this clear separation between conceptual and computerized models has many advantages: it allows designers to describe correctly the system under study using the System Modelling Theory and using the good level of abstraction. Indeed, multiple DEVS formalism extensions and subclasses have been developped (Giambiasi and Carmona 2006, Giambiasi 2009, Hwang 2011, Hwang 2014) in a hierarchical way. Each extension(resp. subclass) encapsulates(resp. is encapsulated in) another formalism. Consequently, the modelling power of DEVS, meaning the level of abstraction, increases or decreases depending on the chosen formalism (Figure 3).

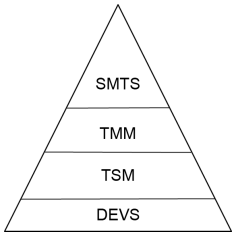


Figure 3: An example of DEVS formalisms hierarchy (Giambiasi 2009).

Moreover, the hierarchical construction of these formalisms means that any model expressed in one formalism can be translated to a DEVS model, and each combination of DEVS model is a DEVS model thanks to the closure under coupling property (Zeigler 1976). This property enables the interoperability between DEVS-compliant simulators. While it is proved that continuous model can also be encapsulated in DEVS model, it allows also the possibility to mix different paradigms in an heterogeneous simulator.

Some attempts to use the DEVS formalism in the case of MANET simulation have been done. Especially, Kim et al. (2007), Tüncel et al. (2016) proposed fundations for using the DEVS formalism as a basis of MANET simulation. These work show that it is possible to take benefits from the advantages of the DEVS methodology, and that is possible to model scalable, adaptive, reusable, costless and powerful mobile network applications. In the first article, the authors propose to use existing MANET simulators like NS2 for modelling low-level network protocols and components, while a DEVS simulator is used as a controller for high-level behaviours and as a handler of interactions between actors and components. However, this architecture always suffer from the lack of precision of the high-level layer and continue to focus on protocol evaluation. Nevertheless, a main idea raises from these experimentations: it is possible to create an interoperability between a MANET simulator and a DEVS simulator. In the second one, the proposed architecture shows that a full-DEVS simulator can easily simulate MANET protocol as accurate as a network-specific simulator, even using a topology generator.

2.3 Towards Integrating Non-DEVS Simulator in DEVS-based Architecture

This last statement is reinforced by the development of a standardized methodology for creating DEVS-based heterogenenous simulation framework (Yong Jae Kim and Tag Gon Kim 1998, Kim et al. 2003). Heterogeneous simulation concerns the use of a collection of simulators developed in different simulation languages and environments, and paradigms, and which work in an interoperable way to achieve a global simulation. A such interoperation needs data exchange and time synchronization between the simulators. While data exchange can be easily resolved through a standard messaging protocol between the simulators, time synchronization is hard because of the possible different natures of the internal models: untimed, continuous, discrete-time or discrete-event. Errors can also come from implementation language which can strongly affects time representation. Parallel and distributed simulations on heterogeneous hardware architecture can also bring errors of approximations.

Considering the universality of the DEVS methodology, Yong Jae Kim and Tag Gon Kim (1998) developed a DEVS-Bus with the idea that it will provide an unified simulation protocol based on DEVS (Figure 4). Each simulator is associated to a protocol converter which transforms this simulator into a DEVS-compliant simulator (Definition 1).

Definition 1. *A DEVS-compliant simulator is a model whose the conceptual model can be defined using a DEVS algebraic formalism, and whose the computerized model follows the DEVS abstract simulator algorithm.*

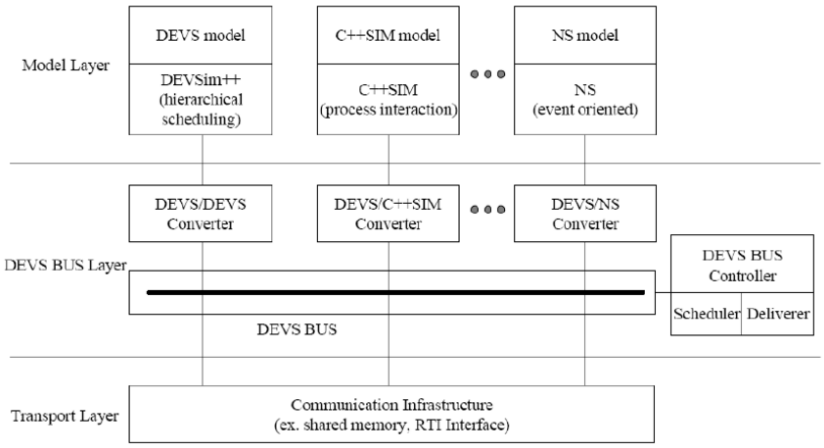


Figure 4: The DEVS Bus developed by Yong Jae Kim and Tag Gon Kim (1998).

Therefore, the set of a non-DEVS simulator and its protocol converter implements a DEVS model, which can be coupled with another DEVS model. The entire coupling becomes a DEVS model. Any kind of simulator can then potentially interoperate with any other kind of simulator with a small overhead, if the protocol converter is well-defined and well-implemented. The challenge is then to define a good converter for each simulation integrated protocol. In the case of MANET discrete-event simulator, the task of protocol definition is easy while both DEVS and MANET simulators use the same simulation paradigm. Therefore, the discrete-event structure of the MANET simulator can be coupled to another DEVS simulator, which can be an heterogeneous simulator which uses the DEVS Bus concept. However, the existing approach lacks of proof of correctness and creates a shift between the network topology and the simulation topology. Indeed, the topology of the network communication doesn't necessarily correspond to the structure of the simulation (i.e. the structure of the coupling), since the coupling corresponds to the communication structure between simulators. This can lead to an important overhead while it becomes impossible to evaluate the performance of applications based on the MANET topology under study.

Taking into account these statements, and the fact that advanced and well-known simulators use a Discrete-Event Architecture, we propose in the next section to integrate them into a DEVS-compliant simulator. We show this approach can be extended to any other MANET simulators while they respect the DEVS principles. Our approach fulfills the following goals adressed by the literature in order to fit to the needs of emergency response simulation:

- Using existing MANET models to achieve MANET-based simulation without recreating specific models for our implementation;
- Taking into account the specificities of each component of the infrastructure by achieving heterogeneous simulation;
- Improving the accuracy of the simulation by executing real software during the simulation and final scenarios, and validating both our simulation model and infrastructure using test cases and use cases;
- Allowing swap between Simulation, Emulation, and Execution without redesigning and redeveloping software thanks to a common interface;
- Allowing the choice of the good level of abstraction while developing the simulator;
- Bringing strong basis for V&V of MANET simulation models and for V&V of the resulted infrastructure.

3 INTEGRATED ARCHITECTURE FOR VERIFICATION AND VALIDATION OF MANET-BASED INFRASTRUCTURE

Our approach will be based especially on the following statements: a DEVS-compliant simulator can be mixed with any DEVS-compliant simulator, and almost everything can be approximated by a DEVS-compliant simulator. First, we define exactly what is a concrete MANET in our case and how it is realized. Then, we show how it can be encapsulated in a DEVS-Bus, and how simulation is finally performed using MANET simulation for the physical part, and the software implementation for the logical part, in order to fulfill our objective to provide an environment which makes possible the verification and the validation of devices and software when they communicate through a MANET.

3.1 MANET Middleware and Heterogeneous Network Stack

Basically, the role of a MANET Middleware is similar to the role of a VPN Middleware:

- Handle and maintain connections between nodes;
- Compute routes between nodes;
- Establish and handle communication steps between nodes.

Therefore, a MANET middleware can be implemented in multiple ways: virtual stack above the OS stack, bridge between layers of the OS stack, etc. However, in an heterogeneous embedded network, there is as many implementations of the OSI stack as there are devices. We would need as many models of the OS stack as there are operating systems. Consequently, our proposed architecture relies on a Virtual Communication Stack (VCS) (Figure 5) which acts as a proxy and hides specific implementation. The VCS hold the following properties:

1. the traffic can goes through the entire stack or can be directly routed to the corresponding OS layer. For the end-user application, data transmission is entirely transparent;
2. the VCS follows a discrete-event architecture;
3. the VCS can be embedded in one or several services which are installed on each device which wants to access the network;
4. each service can be distributed over all the devices (meaning that the implementation of the network can differ according to the platform and the nature of the device);
5. communication is carried over virtual sockets which acts as normal sockets.

Once the VCS runs on a device, communication are done through the OS stack and through the VCS. Indeed, on the one hand, when a message is received by the OS, it is routed to the VCS which dispatch the message to the corresponding application. On the other hand, when an application sends a message, the message goes through the VCS in order to compute the receiver, before going in the OS stack to be sent through the network. Therefore, our simulation environment has to integrate at least two network stack: one to simulate the OS stack, and one to simulate the virtual stack. Considering all the possible configurations, all the possible implementations on different hardware, and all possible physical phenomena, modelling these two stacks represents a great challenge. Instead of modelling the middleware, we take advantage of the DEVS-Bus architecture.

Proposition 1. *Given the automata of a middleware application, we can define a DEVS atomic model which is exactly this automata.*

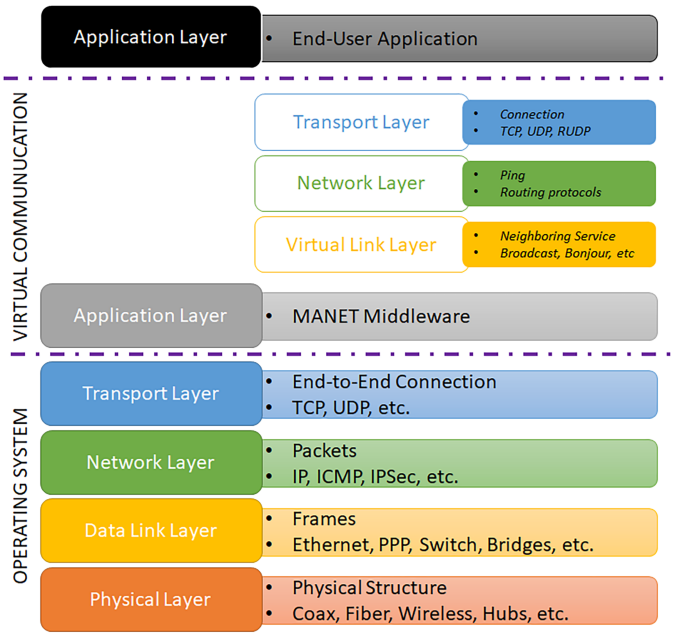


Figure 5: The VCS is between the End-User Application and the OS Stack. The VCS acts as a proxy depending on the execution/simulation mode.

Intuitively, at a high level of abstraction, we can define a conceptual model of a communication middleware as an automata with synchronization mechanisms. The VCS is a discrete-event model by nature (Property 2), with three kind of events: sending, receiving and updating the routing table. At the lowest level of abstraction (the code level), it is a program, i.e. a finite state machine in which each instruction are executed sequentially. Between two executions, the application remains stable. Date of events are decided by the OS scheduler or the CPU clock. In other words, at the lowest level, a middleware is already at least a discrete-time model of our application, and therefore can be see as a discrete-event model. We can also demonstrate this structure is really a system by proving the legitimacy property (Zeigler et al. 2000). However, we can also understand it intuitively with the hierarchy of formalisms. As a consequence, it is not necessary to transform our middleware into another DEVS model, while writing a DEVS proxy is only needed.

Proposition 2. *Given the automata of the OS, we can define a DEVS atomic model whis is exactly this automata.*

The explanation is the same as previously. As a consequence, while the virtual stack calls the OS stack, we can see both of them as a monolithic DEVS model.

A first interesting result appears: while the both stack are already a DEVS model, we can see the whole as a virtual DEVS machine. More precisely, if we can redirect the traffic using a TAP/TUN bridge, therefore it is possible to analyze precisely the communication using the real software instead of a model of the client application. This is close to the TAPBridge functionnality proposed by ns3. Otherwise, we can also make abstraction of the OS layer by modelling it and replacing it by a DEVS model. The choice of the level of abstraction can be done transparently according to the desired configuration.

3.2 Integration of MANET Simulator and Interoperability

The second part of our architecture concerns the integration of existing MANET simulator in order to simulate the physical layer or the OS Stack. For that, we implement two DEVS-Bus (Figure 6):

1. The first one translates and synchronizes event from the VCS. In simulation mode, output are filtered according to time. In emulation mode, output are not modified (they are the result of the execution of the algorithm).
2. The second one translate and synchronizes the input and output of the integrated MANET simulator, in order to make it communicate with the VCS. In emulation mode, this bus send datas directly to the physical media. In simulation, data are sent to the corresponding simulaor.

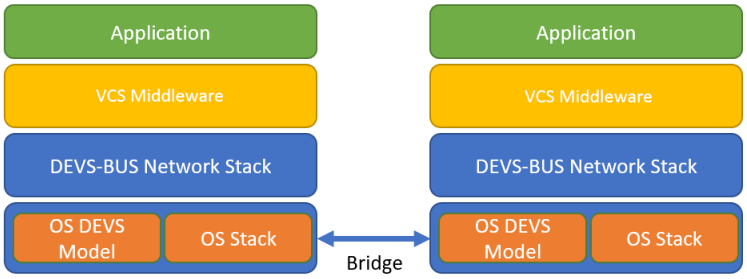


Figure 6: VCS Simulation Environment.

With this modification, communication are modified according to the environment. Therefore, modellers can choose to test their application in execution, emulation or simulation mode. In the first one, the simulators are disabled. The application runs as in operational conditions. In simulation mode, the DEVS-Bus updates the VCS and consequently the end-user application according to the event of the underlying simulator.

3.3 Integrated Software-in-the-Loop and Hardware-in-the-Loop Paradigms

This aspect of our proposed architecture is the use of software-in-the-loop (SIL) and hardware-in-the-loop (HIL) tests. SIL is basically the use of a software instead of a piece of hardware in the test. It is a basic simulation as presented until now or a complete emulation. In contrast, HIL consists on using real piece of hardware in a closed simulation environment. In both of cases, some parts of the architecture can work in *executed time* while others work in *simulated time*. This mainly leads to a synchronization problem: the date of the next event (for instance, $d = 10ms$) is long after the time needed to compute it ($t = 3ms$). To overcome this problem, we implement a controller in the Testing Environment (SIL/HIL). This controller slows down and bufferize the events and transmit them at the correct date to the hardware. In this way, our simulation environment can embbed real piece of software or real hardware, and work in a mixed mode. This allows us to compare model of our physical part (software or devices) with their real implementation without reimplementing all the architecture.

From the point the of view of a end-user application, the communication is entirely transparent. Normal application can then be used to test all the protocols of the network, and then compared with the result obtained in real conditions in order to perform triple validation: validation of the simulation model, validation of the MANET-based infrastructure, and validation of the application.

3.4 Verification and Validation of MANET-based Infrastructure using Integrated Simulation

Another advantage of such an architecture is that we can decide at any steps of the simulation development if we want to use the final software, or a model of this software. If the software is developed using a verifiable and simulable formalism, we can easily ensure its accuracy in the good context. For instance, it is possible to take advantages of the features of combined V&V processes. Indeed, DEVS modelling procedure involves classical V&V processes as defined in (Sargent 2001, Institute 2004) both for conceptual and simulation model. Moreover, if it was proved a general DEVS model cannot be formally verified (Dacharry and Giambiasi 2005, Dacharry and Giambiasi 2007), some subclasses of DEVS models can be translated into formal models (Hwang and Zeigler 2006, Hwang and Zeigler 2009) on which model-checking can be applied. On the another hand, Yacoub et al. (2016) shows that formal models can be combined with DEVS models to take advantages of formal methods and simulation. In this case, a formal model can be translated and augmented with a simulation model. The new obtained model can be then verified and validated using formal verification and formal validation for static properties, and using simulation for checking dynamic behaviours. Integrated in a new software development life cycle, this statement increases the accuracy of the model and of the final software. Indeed, if a final software is built from a verified and validated conceptual and simulation model, it will work as expected designed and tested. Moreover, the capability of using formal methods increases the degree of confidence while the entire statespace can be explored, at least for time-independant properties and invariants. Simulation scenarios can also be easily defined through another External Co-simulation Environment, and integrates more aspects than only the protocol or network parameters.

4 CONCLUSION

Disastrous events generally lead to the deployment of reliable network which will be used by heterogeneous systems and software in order to achieve together critical missions. Simulation of MANETs is used to make these networks reliable but actual research focus only on the accuracy of physical and network models, without taking into account the complexity of application, devices and systems. As a result, they lead to erroneous outcomes and the designed network becomes inoperative in real conditions. We propose some insights to build a new simulation architecture based on the DEVS theory in which MANETs simulation can be used jointly with external simulators, SIL and HIL paradigms in order to answer four main problems adressed in the litterature:

- the granularity, meaning the choice of the right level of details, through the VCS;
- the accuracy implied by the abstraction of the physical model, through the SIL/HIL Testing Environment;
- the V&V of the simulation model during the development process, through the properties of DEVS models;
- the use of realistic cases of application when developing MANET-based infrastructure, through the fact that real application can be used in simulation.

In our approach, models and real systems can be swapped in a trasparent co-simulation environment and adapted at each step of the development of the simulator. The MANETs simulator acts as an oracle which regulates the communication between the different systems. Therefore, at the first stage of simulation development, simplified model can be used to tune different parameters and ensure the network model works properly. Then, real software and hardware can be integrated in the simulator to check their real behaviour, reducing the bias introduced by the software and hardware models. However, if this approach allows modularity, constraints induced by the DEVS theory can lead to major overload. Indeed, our proposed architecture

implies that the network topology corresponds to the coupling topology. However, Classic DEVS doesn't allow removing or adding node during the simulation. Some workaround can be found, but complex mobility with a lot of topological changes is a problem which must be adressed in a future work.

ACKNOWLEDGMENTS

I thank Alexy Torres, and Julien Carayol (Polytechnique Montreal) for their feedbacks that improve the manuscript.

REFERENCES

Andel, T. R., and A. Yasinsac. 2006, July. "On the credibility of manet simulations". *Computer* vol. 39 (7), pp. 48–54.

Cavin, D., Y. Sasson, and A. Schiper. 2002. "On the Accuracy of MANET Simulators". In *Proceedings of the Second ACM International Workshop on Principles of Mobile Computing*, POMC '02, pp. 38–43. New York, NY, USA, ACM.

Chengetanai, G., and G. B. O'Reilly. 2015, March. "Survey on simulation tools for wireless mobile ad hoc networks". In *2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, pp. 1–7.

Dacharry, H., and N. Giambiasi. 2005. "Formal Verification with Timed Automata and DEVS Models: a case study". In *Proceedings of Argentine Symposium on Software Engineering*, pp. 251–265.

Dacharry, H. P., and N. Giambiasi. 2007. "A Formal Verification Approach for DEVS". In *Proceedings of the 2007 Summer Computer Simulation Conference*, SCSC '07, pp. 312–319, Society for Computer Simulation International.

Dorathy, I., and M. Chandrasekaran. 2019, 06. "Simulation tools for mobile ad hoc networks: a survey". *Journal of Applied Research and Technology* vol. 16, pp. 437–445.

Flynn, J., H. Tewari, and D. O'Mahony. 2001. "JEmu: A real time emulation system for mobile ad hoc networks". In *Proceedings of the first joint IEI/IEE symposium on telecommunications systems research*, pp. 262–267.

Frey, H., D. Görgen, J. K. Lehnert, and P. Sturm. 2004. "A Java-Based Uniform Workbench for Simulating and Executing Distributed Mobile Applications". In *Scientific Engineering of Distributed Java Applications*, edited by N. Guelfi, E. Astesiano, and G. Reggio, pp. 116–127. Berlin, Heidelberg, Springer Berlin Heidelberg.

Giambiasi, N. 2009. "From Sequential Machines to DEVS Formalism". In *Proceedings of the 2009 Summer Computer Simulation Conference*, SCSC '09, pp. 216–222. Vista, CA, Society for Modeling; Simulation International.

Giambiasi, N., and J.-C. Carmona. 2006. "Generalized discrete event abstraction of continuous systems: {GDEVS} formalism". *Simulation Modelling Practice and Theory* vol. 14 (1), pp. 47 – 70.

Gomes, C., C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe. 2018, May. "Co-Simulation: A Survey". *ACM Comput. Surv.* vol. 51 (3), pp. 49:1–49:33.

Gunes, M., M. Wenig, and A. Zimmermann. 2007, July. "Improving MANET Simulation Results - Deploying Realistic Mobility and Radio Wave Propagation Models". In *2007 12th IEEE Symposium on Computers and Communications*, pp. 39–44.

Hogie, L., P. Bouvry, and F. Guinand. 2006. "An Overview of MANETs Simulation". *Electron. Notes Theor. Comput. Sci.* vol. 150 (1), pp. 81–101.

Hwang, M. H. 2011. "Taxonomy of DEVS Subclasses for Standardization". In *Proceedings of the 2011 Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium*, TMS-DEVS '11, pp. 152–159, Society for Computer Simulation International.

Hwang, M. H. 2014. "Taxonomy of DEVS Variants". In *Proceedings of the Symposium on Theory of Modeling & Simulation - DEVS Integrative*, DEVS '14, pp. 22:1–22:6, Society for Computer Simulation International.

Hwang, M. H., and B. P. Zeigler. 2006. "A reachable graph of finite and deterministic DEVS networks". *SIMULATION SERIES* vol. 38 (1), pp. 48.

Hwang, M. H., and B. P. Zeigler. 2009. "Reachability Graph of Finite and Deterministic DEVS Networks". *IEEE Transactions on Automation Science and Engineering* vol. 6 (3), pp. 468–478.

Institute, P. M. 2004. *A Guide To The Project Management Body Of Knowledge (PMBOK Guides)*. Project Management Institute.

Ivanic, N., B. Rivera, and B. Adamson. 2009, Oct. "Mobile Ad Hoc Network emulation environment". In *MILCOM 2009 - 2009 IEEE Military Communications Conference*, pp. 1–6.

Johnson, D. B., D. A. Maltz, and J. Broch. 2001. "Ad Hoc Networking". Chapter DSR: The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks, pp. pp. 139–172. Boston, MA, USA, Addison-Wesley Longman Publishing Co., Inc.

Khairnar, V. D., and S. N. Pradhan. 2011, March. "Mobility models for Vehicular Ad-hoc Network simulation". In *2011 IEEE Symposium on Computers Informatics*, pp. 460–465.

Kiess, W., and M. Mauve. 2007. "A survey on real-world implementations of mobile ad-hoc networks". *Ad Hoc Networks* vol. 5 (3), pp. 324 – 339.

Kim, T., M. H. Hwang, D. Kim, and B. P. Zeigler. 2007. "DEVS/NS-2 Environment: Integrated Tool for Efficient Networks Modeling and Simulation". In *Proceedings of the 2007 Spring Simulation Multiconference - Volume 2*, SpringSim '07, pp. 219–226. San Diego, CA, USA, Society for Computer Simulation International.

Kim, Y. J., J. H. Kim, and T. G. Kim. 2003. "Heterogeneous Simulation Framework Using DEVS BUS". *SIMULATION* vol. 79 (1), pp. 3–18.

Klein, Michael 2003. "Dianemu: A java based generic simulation environment for distributed protocols".

Kurkowski, S., T. Camp, and M. Colagrosso. 2005, October. "MANET Simulation Studies: The Incredibles". *SIGMOBILE Mob. Comput. Commun. Rev.* vol. 9 (4), pp. 50–61.

Mallapur, S. V., and S. R. Patil. 2012. "Survey on simulation tools for mobile ad-hoc networks". *International Journal of Computer Networks and Wireless Communications (IJCNWC)* vol. 2 (2).

Manpreet, and J. Malhotra. 2014, Nov. "A survey on MANET simulation tools". In *2014 Innovative Applications of Computational Intelligence on Power, Energy and Controls with their impact on Humanity (CIPECH)*, pp. 495–498.

Mohammed, A., and A. Al-Ghrairi. 2019, 08. "DIFFERENCES BETWEEN AD HOC NETWORKS AND MOBILE AD HOC NETWORKS: A SURVEY". *Xinan Jiaotong Daxue Xuebao/Journal of Southwest Jiaotong University* vol. 54, pp. 12.

Muchtar, F., A. H. Abdullah, M. S. A. Latiff, S. Hassan, M. H. A. Wahab, and G. Abdul-Salaam. 2018. "A technical review of MANET testbed using mobile robot technology". In *Journal of Physics: Conference Series*, Volume 1049, pp. 012001. IOP Publishing.

Murray-Smith, D. 2012. *Modelling and Simulation of Integrated Systems in Engineering: Issues of Methodology, Quality, Testing and Application*. Woodhead Publishing, Limited.

Perkins, C. E., and E. M. Royer. 1999, Feb. "Ad-hoc on-demand distance vector routing". In *Proceedings WMCSA'99. Second IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90–100.

Phillips-Wren, G., and L. Jain. 2006. "Artificial Intelligence for Decision Making". In *Knowledge-Based Intelligent Information and Engineering Systems*, edited by B. Gabrys, R. J. Howlett, and L. C. Jain, pp. 531–536. Berlin, Heidelberg, Springer Berlin Heidelberg.

Reina, D. G., M. Askalani, S. L. Toral, F. Barrero, E. Asimakopoulou, and N. Bessis. 2015. "A Survey on Multihop Ad Hoc Networks for Disaster Response Scenarios". *International Journal of Distributed Sensor Networks* vol. 11 (10), pp. 647037.

Riley, G. F., and T. R. Henderson. 2010. *The ns-3 Network Simulator*, pp. 15–34. Berlin, Heidelberg, Springer Berlin Heidelberg.

Sargent, R. G. 2001. "Some approaches and paradigms for verifying and validating simulation models". In *Simulation Conference, 2001. Proceedings of the Winter*, Volume 1, pp. 106–114.

Sargent, R. G. 2011. "Verification and Validation of Simulation Models". In *Proceedings of the Winter Simulation Conference*, WSC '11, pp. 183–198, Winter Simulation Conference.

Schindelhauer, C., T. Lukovszki, S. Rührup, and K. Volbert. 2003. "Worst Case Mobility in Ad Hoc Networks". In *Proceedings of the Fifteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, SPAA '03, pp. 230–239. New York, NY, USA, ACM.

Schmitz, A., and M. Wenig. 2006. "The Effect of the Radio Wave Propagation Model in Mobile Ad Hoc Networks". In *Proceedings of the 9th ACM International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems*, MSWiM '06, pp. 61–67. New York, NY, USA, ACM.

Sichitiu, M. L. 2009. *Mobility Models for Ad Hoc Networks*, pp. 237–254. London, Springer London.

Takai, M., J. Martin, and R. Bagrodia. 2001. "Effects of Wireless Physical Layer Modeling in Mobile Ad Hoc Networks". In *Proceedings of the 2Nd ACM International Symposium on Mobile Ad Hoc Networking &Amp; Computing*, MobiHoc '01, pp. 87–94. New York, NY, USA, ACM.

Tüncel, S., H. Ekiz, and A. Zengin. 2016. "Design and implementation of a new MANET simulator model for AODV simulation". Volume 24.

Varga, A., and R. Hornig. 2008. "An Overview of the OMNeT++ Simulation Environment". In *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, Simutools '08, pp. 60:1–60:10. ICST, Brussels, Belgium, Belgium, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

Vaubourg, J., V. Chevrier, and L. Ciarletta. 2015, March. "Intégration de simulateurs existants à une plateforme de co-simulation basée sur DEVS". Research report, Loria & Inria Grand Est ; CNRS ; Université de Lorraine.

Yacoub, A., M. e.-a. Hamri, and C. Frydman. 2016. "Using DEv-PROMELA for Modelling and Verification of Software". In *Proceedings of the 2016 Annual ACM Conference on SIGSIM Principles of Advanced Discrete Simulation*, SIGSIM-PADS '16, pp. 245–253, ACM.

Yong Jae Kim, and Tag Gon Kim. 1998, Dec. "A heterogeneous simulation framework based on the DEVS BUS and the high level architecture". In *1998 Winter Simulation Conference. Proceedings (Cat. No.98CH36274)*, Volume 1, pp. 421–428 vol.1.

Zeigler, B. P. 1976. *Theory of Modeling and Simulation*. John Wiley.

Zeigler, B. P., T. G. Kim, and H. Praehofer. 2000. *Theory of Modeling and Simulation*. 2nd ed. Orlando, FL, USA, Academic Press, Inc.

Zeigler, B. P., A. Muzy, and E. Kofman. 2019. *Theory of Modeling and Simulation*. 3rd ed. Academic Press.