APPLICATION-TRANSPARENT INTEGRATION OF SIMULATION TOOLS IN A WSN DEVELOPMENT ENVIRONMENT

Juan Luis Font, Pablo Iñigo, Manuel Domínguez, José Luis Sevillano and Daniel Cascado
Department of Computer Technology and Architecture, University of Seville, Seville, Spain
{juanlu, pabloinigo, mdominguez, sevi, danic}@atc.us.es

Keywords: WSN, ns-3, Serial communication, Simulation, Emulation, Sink node.

Abstract: The use of simulation tools in the development of new Wireless Sensor Networks protocols and applications should be accompanied by minimisation of redundant code and work, making it possible to seamlessly run the same code on simulated and real platforms. This work proposes an architecture for a WSN testing platform that integrates simulation tools in an application-transparent way. The platform will be focused on testing the WSN sink node and providing it with both real and simulated data. The novelty of this approach lies in the use of a flexible network simulation tool not focused on a specific network technology, and the use of generic hardware and open source tools.

1 INTRODUCTION

A Wireless Sensor Network consists of a spatially distributed set of nodes that cooperatively monitor physical or environmental conditions. These nodes are equipped with sensing hardware to collect information from the environment, and radio-frequency communication hardware which makes wireless communication possible.There are several constrains related to power consumption and hardware cost translate in the use of inexpensive hardware with low computing power (Gutiérrez et al., 2003). The nodes that act as gateways between the WSN and an external network are identified as "sink nodes". A sink node forwards information from the WSN to the outside so it needs to be connected to both the WSN and the external network.

The development of WSN based applications benefits from the use of simulation tools, (Obaidat and Papadimitriou, 2003) and (Obaidat and Boudriga, 2010) , which reduce development time and cost. Simulation models usually have a high abstraction level so simulation testing results may differ from real-world ones, but they are still useful enough to evaluate and test new network features. Due to the differences between WSN devices and their simulation models, the code for the testing platform may require considerable changes to be able to run on the final platform. The SER (Kiess and Mauve, 2007) approach (Simulation + Emulation + Real-world) aims to integrate simulation, emulation and real-world based experiments while keeping the differences between testing platforms application-transparent for the tested code. Emulation is defined in our context as the use of a combination of real-world and simulated components. Nevertheless, some platform systems has been chosen due to their compliance with other emulation definition where virtual elements mimic their real counterparts functionality in a indistinguishable way.

This work proposes a testing platform built based on simulation tools and real-world WSN elements (Sobeih et al., 2006). It will allow application-transparent integration between WSN elements and simulation tools. The ultimate goal is the definition of a testing platform where real-world elements can be replaced by simulated or emulated ones in a application-transparent way.

2 ARCHITECTURE DESCRIPTION

The architecture identifies the main elements of the platform by defining their roles. The overall structure is shown in the Figure 1.

- WSN is replaced by a simulation. Its mission is to feed the sink node with simulated data.
- The sink node runs on a real hardware platform. Its components are real-world devices with the ex-

ception of the communication channel which connects it with the simulated WSN.

- The link between simulated WSN and sink node will be an emulated communication channel. Both simulation and sink software will be adapted to intercommunicate through this emulated link.

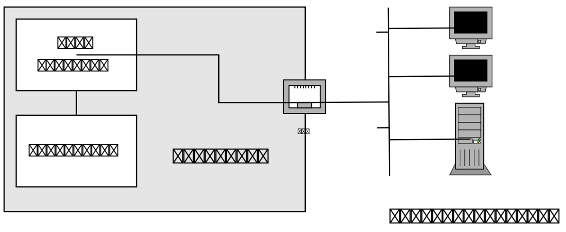
Figure 1: Overall view of the architecture proposal.

The sink node is built based on standard x86 hardware. A standard personal computer provides a hardware platform suitable for developing and testing sink nodes and it has a considerable amount of computing resources in comparison with the average WSN node. This work has used the prototype sink node as host device of the simulation process too, so it has been mandatory to choose a hardware platform powerful enough to simultaneously run both the sink software and the network simulation. The extra resources are aimed to support the simulation execution.

All the sink devices are real with the exception of the WSN network interface. This radio-frequency device is replaced as data source by a simulation process, as shown in Figure 2. Thanks to the proposed design, the simulated data source can be replaced by a real one without a major rewrite of the sink software.

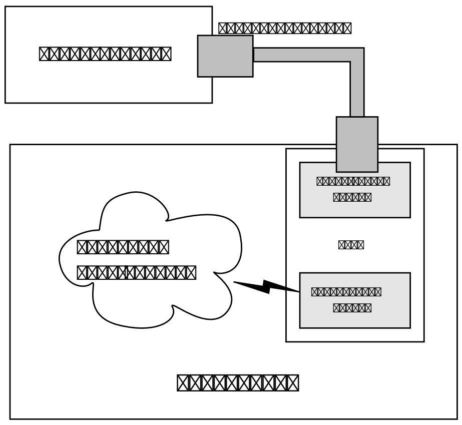
Figure 2: Structure of the simulation elements.

The WSN is replaced by a simulation that generates data and forwards it to the sink node. This first proposal replaces the WSN data with a basic simulation. An existing network simulator has been chosen as the basis for the simulation. This reduces the coding effort to adapt the simulation tool and enable its communication with the sink software. Future improvements can be made in the simulator to provide it with new features and improve its accuracy and functionality. The integration work has focused on enabling the WSN simulation to dump data in the emulated communication channel which interconnects it with the sink software.

The emulated communication channel interconnects the sink software with the simulated WSN. Emulation has been chosen because it is very close to its real-world counterpart in terms of functionality, performance and interface, so the change from this virtual channel to a real one will have little or no effect on the source code communication.

3 TECHNICAL CONSIDERATIONS

This section discusses technical topics related to the testing platform architecture and describes a set of hardware and software elements that have been used for its implementation.

3.1 Network Simulator

Ns-3 is a suitable choice for simulation task in the testing platform (Font et al., 2010), (Font et al., 2011). Ns-3 doesn't have specific models for WSN technologies such as Zigbee or Bluetooth. It can implement general purpose wireless networks based on 802.11 or WiMAX technologies and counts with propagation and physical channel models (Obaidat and Boudriga, 2010). Nevertheless, the aim of the platform is providing data to the sink node. The simulation is focused on generating data traffic and forwarding it to the sink node.

The ns-3 simulation generates data that is forwarded to the node object which contains the P2PEmuDevice object. This relays the information outside the simulation environment through an emulated channel. The process that implements the sink routines only has to connect to the above emulated channel.

Ns-3 already has class to emulate a network device which is internally connected to a real Ethernet device, the `ns3::EmuNetDevice`. The `ns3::P2PEmuDevice` class has been defined following this design but replacing the real Ethernet device by a serial device and adapting it to the requirements of the point-to-point transmission.

The figure 4 shows the inheritance diagram of the P2PEmuDevice class. The `ns3::P2PEmuDevice` in-

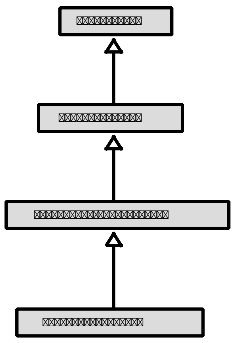

Figure 3: Inheritance diagram of P2PEmuDevice class.

stance is contained in a `ns3::Node` object. The ns-3 static routing mechanisms help to forward data from any other simulated device to the P2PEmuDevice one. This also needs an auxiliary queue with tail drop algorithm, implemented by `ns3::DropTailQueue`.

The use of a real-time simulator is needed to interact with the real world on regular time basis so the simulation makes use of the `ns3::RealTimeSimulationImpl`, also included in ns3.

The Figure 3 shows the relation between P2PEmuDevice and several other ns-3 classes.

3.2 Hardware and Software Platform

Standard hardware based on x86 has been used to build the sink node. A standard personal computer can easily meet all the sink hardware requirements with cheap components and the addition of a radio-frequency network device to allows it to communicate with the rest of the WSN. Some previous gateway proposals have chosen embedded ARM solutions (Song et al., 2008), (Yang and Yan, 2010). Due to their amount of resources, they have even the capacity of running general purpose operating systems or adapted versions such as embedded versions of GNU/Linux or Windows. Moreover, choosing such hardware platforms allows the use of a wide range of development and debugging tools.

Thus, the prototype sink node hardware considerably differs from the rest of the autonomous nodes in terms of hardware, software, power consumption and computing power. The sink node hosts both WSN simulation and sink software.

The platform prototype has been built on Debian GNU/Linux 6.0 Stable, which support ns-3 simulator and it has all the required software dependencies in its package repository. Other general purpose GNU/Linux distributions are also suitable for running ns-3, e.g.. Fedora or Ubuntu.

3.3 Emulated Communication Channel

A pseudo-terminal is defined as pseudo-device pair which provides a text terminal interface without associated virtual console, computer terminal or serial port hardware. Instead, a process acts as the underlying hardware for the pseudo terminal session. It is present in some operating systems, including Unix-like ones and there are several implementation approach and standards that regulate their nomenclature, and programming API. Unix 98 standards define an API to create and use pseudo terminals which are not restricted in number by nomenclature, only by implementation. It proposes a master-slave design based on two different types of pseudo-devices linked between them: Pseudo Terminal Master (PSM) responds to `/dev/ptmx` device name in GNU/Linux systems; and Pseudo Terminal Slave (PTS) have a name of the form `/dev/pts/X` in the GNU/Linux file system, being X a natural number which automatically increases in one unit for each new PTS. Each time a process opens the PTM device, a new PTS is automatically created, existing a connection between the file descriptor of the PTM and the slave pseudo terminal.

Basic Unix 98 API functions have been used to interact with the PTM device:

- `int posix_openpt(int oflag)`, opens the `/dev/ptmx` device and returns the associated file descriptor.
- `int grantpt(int fd)`, grants access to the slave pseudo-terminal.
- `int unlockpt(int fd)`, unlocks a pseudo-terminal master/slave pair.
- `char *ptsname(inf fd)`, gets the name of the pseudo-terminal.

4 CONCLUSIONS AND FUTURE WORK

This work has defined and implemented the architecture of a WSN platform that integrates simulation and real elements in a application-transparent way. The main elements have been identified, their roles have been defined and several technologies and tools have been proposed to implement the testing platform. The resulting proposal lays the groundwork for future works to improve its features and determine its scalability and range of application.

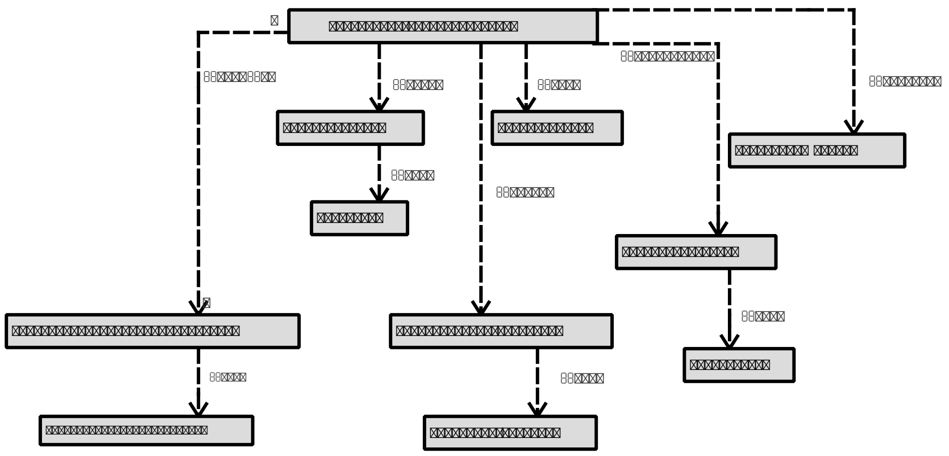

Figure 4: Simplified ns3::P2PEmuDevice class diagram.

There are some topics likely to be discussed and expanded in future related work such as the evaluation of time restrictions, the scalability of the WSN simulation or the overall performance of the platform. Another interesting line of work is laying the foundations for monitoring the system to detect performance degradation. Integration of emulation tools is also proposed to allow simulated, emulated and real-world experiments in the same platform.

ACKNOWLEDGEMENTS

This work was partially supported by contract Vulcano: TEC2009-10639-C04-02.

REFERENCES

Font, J. L., Iñigo, P., Domínguez, M., Sevillano, J. L., and Amaya, C. (2010). Architecture, design and source code comparison of ns-2 and ns-3 network simulators. *CNS-10, 13th Communications & Networking Simulation Symposium*, pages 29–36. Best Paper Award.

Font, J. L., Iñigo, P., Domínguez, M., Sevillano, J. L., and Amaya, C. (2011). Analysis of source code metrics from ns-2 and ns-3 network simulators. *Simulation Modelling Practice and Theory*, 19(5):1330 – 1346.

Gutiérrez, J. A., Callaway, E. H., and Barrett, R. L. (2003). *Low-Rate Wireless Personal Area Networks*. IEEE Press, New York, 1st edition.

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

Obaidat, M. S. and Boudriga, N. A. (2010). *Fundamentals of Performance Evaluation of Computer and Telecommunication Systems*. John Wiley & Sons, Hoboken, New Jersey, 1st edition.

Obaidat, M. S. and Papadimitriou, G. (2003). *Applied System Simulation: Methodologies and Applications*. Springer, 1st edition.

Sobeih, A., Hou, J. C., Kung, L., Li, N., and Zhang, H. (2006). J-sim: A simulation and emulation environment for wireless sensors networks. *Wireless Communications, IEEE*, 13(4):104–119.

Song, P., Chen, C., Li, K., and Sui, L. (2008). The design and realization of embedded gateway based on wsn. *International Conference on Computer Science and Software Engineering*, 4:32–36.

Yang, F. and Yan, C. (2010). Design of wsn gateway based on zigbee and td. *ICEIE 2010, International Conference on Electronics and Information Engineering*, 2:V2–76–V2–80.