DIPARTIMENTO DI INFORMATICA
E SISTEMISTICA ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Simulation vs real testbeds: a validation of WSN simulators

Lorenzo Bergamini
Carlo Crociani
Andrea Vitaletti

Technical Report n. 3, 2009

DIPARTIMENTO DI INFORMATICA
E SISTEMISTICA ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Simulation vs real testbeds: a validation of WSN simulators

Lorenzo Bergamini
Carlo Crociani
Andrea Vitaletti

Technical Report n. 3, 2009

Simulation vs real testbeds: a validation of WSN simulators

Lorenzo Bergamini
DIS - Sapienza Università di Roma
via Ariosto 25, 00198 Roma, Italy
Email: bergamini@dis.uniroma1.it

Carlo Crociani
DIS - Sapienza Università di Roma
via Ariosto 25, 00198 Roma, Italy
Email: crociani@dis.uniroma1.it

Andrea Vitaletti
DIS - Sapienza Università di Roma
via Ariosto 25, 00198 Roma, Italy
Email: vitaletti@dis.uniroma1.it

March 2, 2009

Abstract

In [9] we presented a first effort in assessing the reliability of OMNeT++ in the simulation of Wireless Sensor Networks. We showed that OMNeT++ with MACSimulator framework over-estimates the experimental results obtained by a real testbed. In this paper, we extend our analysis to the well known NS-2 simulator and then to Castalia, a new framework to simulate WSNs on OMNeT++. We first describe the architectures of the two simulators and we analyze how well they model the general structure of a WSN node, mainly in terms of application, MAC, radio and physical layers. We then compare testbed results obtained on several network topologies to the simulation results on the same topologies and we observe that both simulators provide quite reliable results (approximately 10%-30% over/under-estimation of experimental results). In our simulations we exploited both the flat disk propagation model and the log-normal shadowing path loss model and, surprisingly, we observed that the simple flat disk model produces more reliable results on dense topologies.

Keywords Wireless Sensor Networks, Simulation environments, Performance evaluation, Reliability

1 Introduction

Wireless Sensor Networks (WSNs) are still in a pre-industrial phase, but the continuous advancements in microelectronics are paving the way for their future large scale deployment. Nevertheless, many of the logistical challenges related to the development, deployment, and debugging of realistic large-scale sensor networks have gone unmet: manually reprogramming nodes, deploying them into the physical environment, and instrumenting them for data gathering is tedious and time-consuming. Furthermore, once a sensor network is deployed, accessibility to nodes forming the network drops dramatically and debugging becomes hard; the limited communication and computational resources prevent nodes from freely storing and transmitting debugging information, as this quickly depletes energy and consequently decreases network lifetime.

Due to the complexity and difficulty to implement real testbeds, simulators are widely used, and indeed in the vast majority of the papers on algorithms and protocols for wireless sensor networks, the performance evaluation is mainly based on simulation results. Simulations allow researchers to validate WSN solutions before deployment, so as to reduce the need for corrective actions once the network is actually operating, and furthermore they enable the large scale experimentation of protocols and applications in a flexible and scalable environment.

Testbeds and simulators are two important and complementary design and validation tools; an ideal development process should start from the theoretical analysis of the protocol providing bounds and indication of its performance, verified and refined by simualtions and finally confirmed in a testbed. Even if network simulators are the preferred validation tool for WSNs, their reliability was only marginally investigated. In [9] we have shown that even in a simple scenario where two nodes are connected and both flood the network, the distance between testbed and simulation metrics can be significant.

Contribution of the paper In this paper we adopt the same approach of [9], namely we compare the simulation results to the results of a real WSN testbed to obtain a clear assessment of the reliability of WSN simulators. We extended the performance evaluation to the well known network simulator NS-2 [1] and we consider the new Castalia framework [3] for OMNeT++. Both these simulation tools provide more accurate hardware, protocol and propagation models than MACSimulator. Furthermore we show that in the considered WSNs scenarios the simplistic flat disk model guarantees the best accuracy; this is partially in contrast with previous results in [4]. In section 2 we provide a comparison of the features offered by NS-2 and Castalia. Castalia is characterized by a very clear modular architecture, which allows users to limit the tuning and code updates to well defined areas of the simulator. On the contrary, NS-2 is characterized by an extremely steep learning curve and any code update is burden by a myriad of inter-dependences between the C++ classes forming the NS-2 core. In section 3 we describe our experiments and we introduce the metrics we used to compare simulations and testbeds. In section 4 we evaluate the reliability of the results provided by NS-2 and Castalia by measur-

ing the distance between the values of performance metrics of the two simulators on a particular network scenario and those obtained on a real testbed. Our experiments show that both simulators can guarantee accurate results: simulation metrics are $\pm$ 10% - 30% from real values. Finally, in section 5 we discuss our results and we propose future research work.

2 Castalia and NS-2 as simulation tools for WSNs

An extensive comparison of simulators for WSNs from a functional point of view is provided in [2]. In this paper we focus on two open source simulators, Castalia and NS-2, and we analyze how they are well suited to support the simulation of WSNs.

Castalia is a WSN simulator based on the OMNeT++ platform that has been presented to the scientific community July 2007, by the NICTA[1] [3]. The main features offered by Castalia are: advanced channel/radio models based on empirically measured data, detailed state transitions for the radio, multiple transmission power levels, resource monitoring that goes beyond energy consumption (such as memory and CPU time), fine tuning of all parameters involved in communications between MAC and physical layers. NS-2 is probably the most widely adopted network simulator. It allows users to add new protocols, agents or applications to the compiled hierarchy of the simulator. Nevertheless, contrary to Castalia, the structure and boundary of each module are not well defined and characterized by many software inter-dependencies. In fact, any new release of the simulator requires some work for developers to adapt their "old" simulations to the new core software modules. NS-2 was originally mainly designed to simulate TCP/IP networks and thus the implementation of any new protocol stack requires update of some of its functionalities, and re-compile of the whole simulation environment. An attempt to develop an extension specifically suited for WSNs has been made in 2000 by the University of California (Los Angeles) with Sensorsim [8], but unfortunately the project has been abandoned.

A critical job in WSN simulators is modelling the sensor node and in particular the strict correlation, typical of embedded systems, that exists between hardware and software components. In figure 1 the general architecture of a real WSN node is depicted while figure 2 shows how the two simulators model the concept of sensor node. As clearly emerges from the figures, Castalia well reflects the general architecture of a wireless sensor node: the Communication Subsystem is mapped into the *Radio*, *MAC* and *Routing* modules of Castalia, while the Application Layer Subsystem is mapped into the *Application* module.

The same considerations do not apply to NS-2: for example, the radio is not explicitly defined in any module, but its functionalities are performed by the *WirelessPhy* and the *Propagation* modules while the Sensing Subsystem is completely neglected (in any case we are not going to use this subsystem in our experiments).

[1] Australia's centre of excellence in ICT research

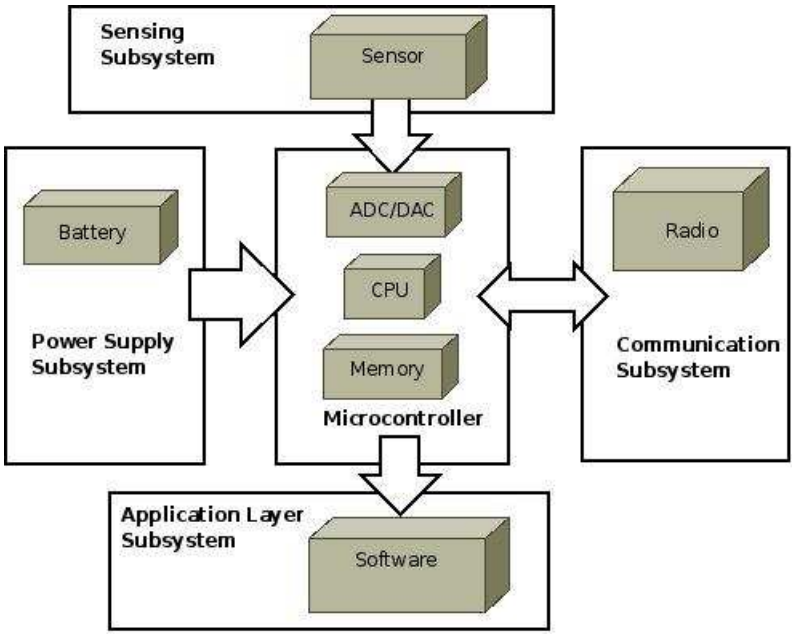

Figure 1: General architecture for a wireless sensor node

In the following we go further in our analysis, extending the comparison between Castalia and NS-2 to all the main components involved in our experiments.

2.1 Modelling WSNs

In our experiments, a WSN can be viewed as a set of *radio* equipped nodes running the *flooding* algorithm at application layer, and accessing the *wireless channel* through the *MAC* protocol. In the following, we discuss how all these components are modelled in NS-2 and Castalia.

Wireless Channel Both NS-2 and Castalia provide the *flat disk propagation model* (i.e. two nodes communicate if their distance is within a fixed communication radius) and the more refined *log-normal shadowing path loss model* described in [7] and characterized by the following formula:

$$PL(d) = PL(d_0) + 10\eta \log_{10}(\frac{d}{d_0}) + \chi_\sigma \quad (1)$$

Where d is the transmitter-receiver distance, d_0 is a reference distance, η is the path loss exponent and χ_σ is a zero-mean Gaussian random variable (in dB), that in the most general case is a random process function of the time, with standard deviation σ (shadowing effects). The received signal strength at a distance d is the output power of the transmitter minus $PL(d)$. When $\sigma = 0$, all nodes within

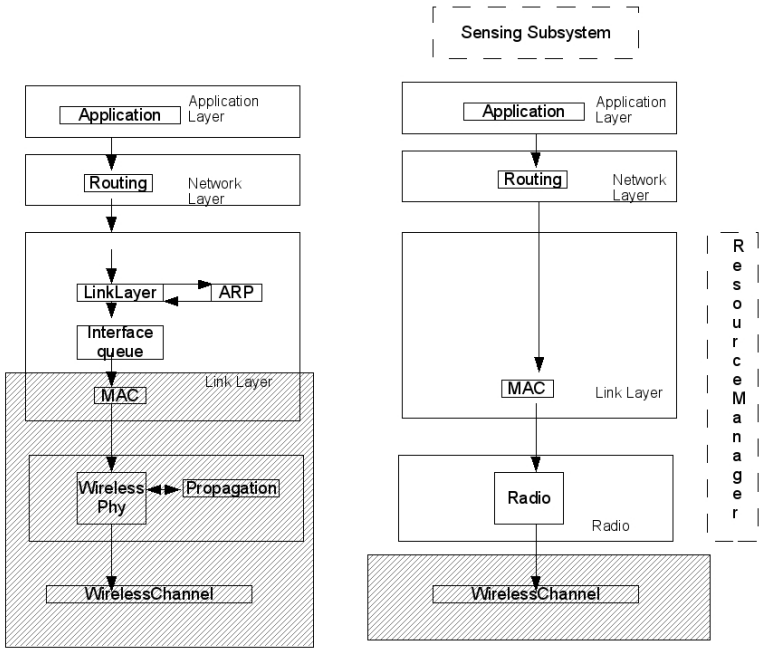

Figure 2: NS-2 node model on the left and Castalia node model on the right

a certain distance from the transmitter get the same signal strength, all links are bidirectional and the model is reduced to the flat unit disk model. Empirical studies have shown that the log-normal shadowing model provides more accurate results for multi-path propagation than Nakagami and Rayleigh [5]. The main difference between NS-2 and Castalia is the computation of $PL(d_0)$: in Castalia it is fixed to 55dBm according to [10], while in NS-2 this value is dynamically computed as function of distance.

A packet is correctly received if the received power is above a certain threshold. The verification of this condition is demanded to the *Wireless Channel* module in Castalia, while in NS-2 the *Propagation* module calculates the received power and the *Wireless Channel* compares it with the threshold. Three main collision models (CM) are implemented in Castalia: *CM=0)* where no collisions happen even if several nodes are concurrently transmitting, *CM=1)* where if a receiver can hear two transmissions concurrently (i.e. they partially overlap) then a collision is assumed, irrespective of their relative strength at the receiver, and *CM=2)* where collisions are calculated considering the SIR (Signal to Interference Ratio). In NS-2 only *CM=2* is currently implemented. In Castalia the occurrence of collisions is verified by the wireless channel, whereas in NS-2 the same function is performed at MAC layer. In principle, collisions occur in the wireless channel and are managed by the MAC layer and in a CSMA/CA (Collision Avoidance) environment they

should not be detected at MAC layer as it happens in NS-2. In table 1 we summarize the value of the parameters used in our simulations; similar parameters hold for NS-2.

Parameters	ideal	realistic	Description
η	2.4	2.4	path loss exponent
$PLd0$	55	55	dBm the Path Loss at $d0$
$d0$	1	1	meters
σ	0	4	$\sigma = 0$ –> flat disk model
allBidirectionalLinks	true	false	has meaning if ($\sigma \neq 0$)
collisionModel (CM)	0	2	Parameters that deal with the realistic features of the channel (channel accuracy) 0 –> No Collisions (no Interference) 2 –> Additive interference model (transmissions seen as interference)

Table 1: Parameters of Castalia wireless channel

Radio In Castalia, the *Radio* module tries to model most of the main features of a generic low power modern communication hardware: it supports carrier sense (querying the wireless channel) and multiple states with different power consumptions and delays for each state transition (e.g. idle to TX). This allows to well model all main radio hardwares such as CC2420 (TMote SKY) and CC1000 (MICA 2).

In NS-2 the radio hardware component is not a unique C++ class. Instead, radio functionalities are performed by the *WirelessPhy* and the *Propagation* classes.

MAC To model our testbed made of Tmote SKY nodes, we used a simple CSMA/CA protocol (i.e. a carrier sense collision avoidance MAC protocol) that has been implemented in Castalia by simply tuning some of the parameters of the built-in TunableMAC protocol.

In NS-2, on the other hand, we wrote the C++ source code from scratch implementing the CSMA/CA protocol. Furthermore, as observed before, in NS-2 there is not a clear distinction between MAC functionalities and channel functionalities.

Application Layer (flooding) In Castalia the application layer is built on top of the communication stack: we simply connected the stand-alone module implementing the flooding algorithm (see figure 1) to the MAC module. The flooding module is responsible for the generation of messages and for their propagation over the whole network. In NS-2 packets (i.e. messages) generation and consumption is managed by an *agent* initialized and controlled by an OTcl script. Thus the concept of application layer in the protocol stack is not explicitly modeled.

3 Experiment Set-up and Metrics

To evaluate the reliability of Castalia and NS-2 we walk through the following simple steps:

1. We record the metrics of the testbed on several scenarios.
2. We simulate the same scenarios and collect metrics from both simulators.
3. We compare the results of points 1 and 2.

Design principles We designed our experiments to be as simple and clear as possible. The rationale behind this choice is that, if a distance between simulation and testbed emerges in a very simple and clear environment, this distance is likely to be further extended by a more complex one. Following this principle, we considered a limited but significant number of nodes (at most 12) connected in a dense topology (i.e. a clique), and running the basic flooding algorithm (see Algorithm 1) over a simple CSMA/CA mac protocol defined tuning the parameters of the Castalia *TunableMAC* according to table 2.

Nevertheless, our results show that the global behavior of a system designed according to this simple principle can be surprisingly complex.

Parameters	Values	Description
dutyCycle[2]	1	listening / (sleeping+listening)
beaconIntervalFraction	0	beacon_interval / sleeping_interval, if 0 no beacons are sent
probTx	1	the probability of a single try of Transmission to happen
reTxInterval	0	(ms), Interval between retransmissions, (numTx-1) retransmissions
backoffBaseValue	0.64	(ms)
backoffMaxValue	10.24	(ms) backoff Value $\in [640, 10240]\mu S$
maxMACFrameSize	37	(bytes)
macFrameOverhead	5	(bytes)
ACKFrameSize	5	(bytes)
macBufferSize	1	number of maximum frames held from the upper layer

Table 2: Parameters of Castalia TunableMAC to simulate BMAC

Metrics In our experiments we focus on the metrics reported in table 3. Metrics are recorded at the end of each simulation for each node. We consider the number of messages sent/received both at application and MAC layers (e.g. a packet transmitted at application layer could be discarded at MAC layer because of contention). Since in our experiments a message is delivered in a single packet, we will use these terms as synonyms.

Although metrics in table 3 allow us to have a clear picture of the behavior of the WSN during communication, a unique simple function calculated as an aggregate of all those metrics would make it easier to homogeneously compare simulators and testbeds in all scenarios considered. This function is defined as follows:

$$X_{Sim}^{Scen} = \frac{(TX_{MAC,APP} + RX_{MAC,APP} + Dup)_{Sim}^{Scen}}{(TX_{MAC,APP} + RX_{MAC,APP} + Dup)_{Tbed}^{Scen}} - 1$$

Algorithm 1 Flooding algorithm

```
1: repeat
2:
3:     if (the node is a generator) then
4:         data ← ADC.data()                          ▷ Get data from sensor
5:         msg ← data
6:         bcast(msg)
7:         num_msg++
8:         wait(I_samp)
9:     end if
10:
11: until (num_msg == MAX_MSG)
12:
13: receive(msg)                                      ▷ All nodes
14: if (¬ Received(msg)) then                         ▷ Received a new msg
15:     store(msg)
16:     bcast(msg)
17:
18: else
19:     discard(msg)
20: end if
```

Metrics	Description
TX_{APP}	Num. of packets sent by the application layer;
RX_{APP}	Num. of packets received by the application layer;
Dup	Num. of duplicates received and consequently discarded by the application layer;
TX_{MAC}	Num. of packets transmitted at MAC layer;
RX_{MAC}	Num. of packets received at MAC layer;

Table 3: Metrics

The more X_{Sim}^{Scen} tends to 1, the better the simulator (Sim) approximates the testbed behavior in the considered scenario ($Scen$), in addition a positive/negative value means that the simulator over/under estimates the testbed metrics.

Wireless Channel Settings All metrics above can be strongly affected by the wireless channel model adopted in the simulations. In this work, we experimented with both the simple flat disk model and the log-normal shadowing path loss model (see section 2.1). In particular, we consider the *ideal* settings (denoted by NS-2 and Castalia) and the *realistic* settings (denoted by NS-2* and Castalia*) as described in table 1. Recall that in NS-2 the collision model $CM = 0$ is not implemented and thus we used the $CM = 2$ also for the ideal case.

Experimental Setup We consider three different network topologies consisting of 2, 6 and 12 nodes at an average distance of 5 meters, always connected in a clique and where at least one and at most all the nodes are generators injecting 200pkts/sec of 42 bytes each. The maximum number of packets generated is set to 18000 for each generator. These are dense networks with a considerable amount of traffic flowing through the nodes; we want to saturate the network to verify if simulators are able to reproduce the real performances in such a critical situation. If so, we can infer that the simulators, in lighter traffic environments, would perform at least as well as the analyzed scenarios. To identify each of the scenarios we use the notation `<i>n<l>g` where `i=2,6,12` is the number of nodes and `<g>` is the number of generators; for example `12n1g` identifies the network made of 12 nodes connected in a clique where 1 node is a generator that floods the network. Metrics on each scenario are calculated as the average of the values collected by each node on 50 independent runs where for each run we changed the seeds used as the starting point by the random number generators.

Testbeds In our testbeds we used the Tmote SKY nodes produced by Moteiv corporation equipped with Boomerang OS, deployed outdoor to minimize the effects of interferences and reflections, and placed on a 40 cm high support to ensure a clear Fresnel Zone.

4 Results

We ran our experiments in all the above described scenarios (i.e. `<i>n<l>g`), but for the sake of space, in this work we mainly focus on two significant cases where: a) there is only one generator, which well describes situations where a node starts a process based on the flooding protocol such as interest dissemination or clock synchronization in a WSN; b) all nodes are generators, as happens in a cluster of nodes all sensing the same phenomenon. In particular we will consider the 2n1g and 2n2g simple scenarios because in [9] we observed that even with these simple

settings the distance between simulation and testbed metrics can already be significant and the 12n1g and 12n12g scenario which can be considered as representative of a dense cluster in a WSN. In all our experiments, NS-2 simulation results do not show any significant difference between the metrics collected using the *ideal* settings and the *realistic* ones. Recall that a packet is correctly received if the receiving power is above a certain threshold. In all the scenarios above the small distance between the nodes (less than 10 metres) and the channel parameters (e.g. path loss exponent) makes the receiving power calculated by NS-2 to be similar in both models, and thus the threshold condition is either satisfied or not independently from the adopted model. For this reason, in the next figures results on NS-2* are not reported; as we will see, this is not the case with Castalia and Castalia*.

2n1g: 2 nodes 1 generator We start our experiments with the most simple scenario consisting of 2 connected nodes and only one generator. On this basic network scenario, we expected all the simulators to produce quite reliable and similar results, but as can be seen in figure 3 this is not the case. Indeed, considering the testbed metrics as reference values (i.e. the unit in figure 3), NS-2 and Castalia both tend to under estimate. For example, receptions at MAC layer RX_{MAC} in NS-2 are only 60% of those in the testbed and Castalia increases this percentage to about 80%. Furthermore, the simplest MACSimulator tool considered in [9], is the more accurate. This is due to the fact that in this scenario, the testbed behavior is pretty similar to the "ideal case", where all the packets are correctly transmitted and received and thus MACSimulator, that neglects any complex aspect of the wireless channel such as propagation, interferences and collisions, produces the best results. Finally Castalia and Castalia* show similar performance and thus we only report the results on the ideal settings (i.e. Castalia).

2n2g: 2 nodes 2 generators The topology of the next scenario we consider is an exact copy of the previous one, but this time both nodes inject packets into the network. Although this seems to be only a little variation, MACSimulator starts to show all its limits and produces extremely bad results; in figure 4, the MACSimulator line is most of the time out of scale. For example, TX_{MAC} in MACSimulator is almost the double of TX_{MAC} in the real testbed. This confirms that MACSimulator's behavior is nearer to the ideal case rather than to a real testbed. Castalia simulation results are again the more accurate, even if in this case they over estimate the performance of the network, and they are still only marginally affected by the wireless channel settings.

12n1g: 12 nodes 1 generator A testbed consisting of 12 nodes where one of them is a generator, can be considered as representative of a cluster in a larger

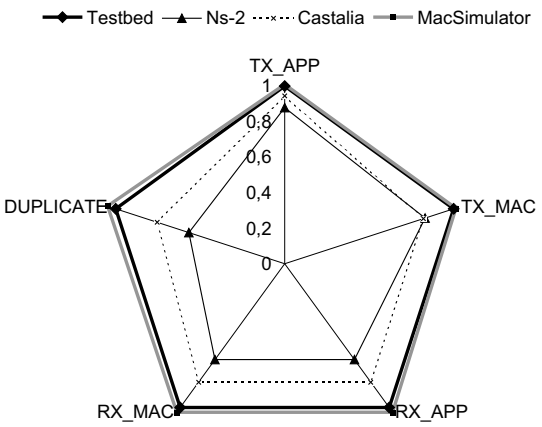

Figure 3: 2 nodes 1 generator

network [3], where the cluster head (i.e. the generator) starts for example a time synchronization process. This is a more complex scenario and for the first time a difference between ideal and realistic settings in Castalia emerges (see figure 5). Surprisingly, the realistic setting (i.e. Castalia*) produces less accurate results. For example, the number of received packets both at application and MAC layers are less than 50% those received in the testbed. The main problem of Castalia* is that the SIR (Signal to Interference Ratio) is always under estimated and this produces an excessive number of discarded packets due to collisions. Noticeably, both Castalia and NS-2 provide quite accurate results even if Castalia is confirmed as best performer. Note that even if the number of transmitted packets (TX_{MAC},TX_{APP}) is similar both in Castalia and in Castalia*, the number of receptions is far lower in Castalia*. This apparently strange phenomenon is absolutely coherent with the flooding protocol we used (see Algorithm 1): every new packet is re-transmitted while duplicates are not, this implies that receiving a packet once or several times does not increase the number of transmitted packets.

12n12g: 12 nodes 12 generators Finally, we consider the same cluster consting of 12 nodes, where all the nodes are generators (e.g. they all sense the same phenomenon). In a clustered network, communications are coordinated by the cluster

[3] According to [6] 12 nodes is the optimal cluster size for a network of 78 nodes.

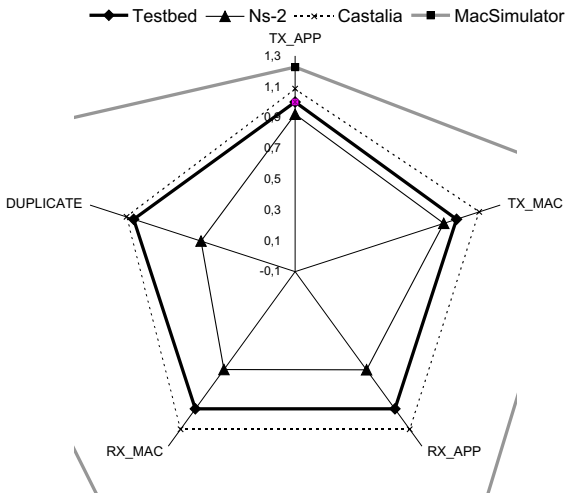

Figure 4: 2 nodes both generators

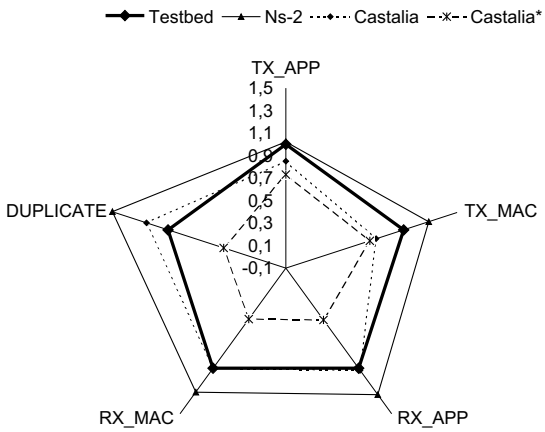

Figure 5: 12 nodes , 1 generator

head, which usually avoids flooding-like mechanisms, but we consider this scenario particularly interesting because it allows us to stress both the simulators and the testbed. Also in this case Castalia* is the worst performer (see figure 6) and Castalia and NS-2 show similar behaviors except for the number of transmitted packets at MAC layer. The number of TX_{APP} is always over-estimated by simulators. In real nodes, every 5ms a new sending task is posted, but the corresponding message at application layer is generated only once the task is actually executed; in other words, the generation of the TX_{APP} is asynchronous. Furthermore, sending tasks can be interrupted by hardware events. This implies that messages can be possibly discarded and are in any case affected by unpredictable delays. In NS-2 these aspects are not well modelled and all messages are delivered to the MAC layer exactly every 5ms. In this scenario, the sending process is stressed both at application and MAC layer; in particular the MAC layer cannot handle the considered message generation rate (i.e. about 5ms) and thus it acts as a "low-pass" filter. Surprisingly, Castalia well capture this behavior and even if the TX_{APP} are over-estimated, the number of TX_{MAC} is very accurate.

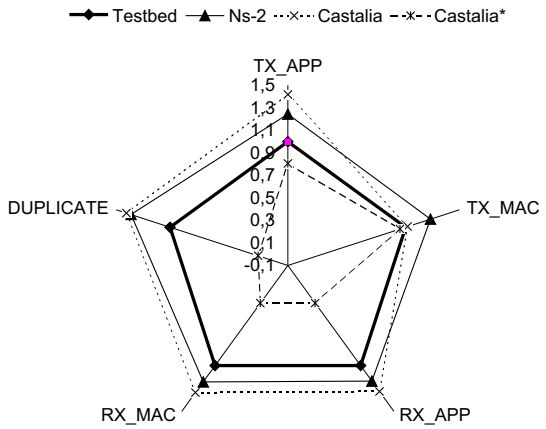

Figure 6: 12 nodes , 12 generators

5 Conclusions

Our experiments show that both Castalia and NS-2 provide quite reliable results on the considered scenarios as can be observed in figure 8, where we report the

value of the function X_{Sim}^{Scen} defined in section 3. Recall that this function allows us to homogeneously compare the performance of simulators and testbeds. Figure 8 confirms that Castalia and NS-2 are the best performers as they over/under-estimate testbed results by at most 30%. Nevertheless, Castalia shows slightly better performance in terms of reliability of the simulation results and substantially better results in terms of efficiency; in our simulations Castalia is two orders of magnitude faster than NS-2. Surprisingly Castalia*, which is supposed to provide the most accurate communication model, is the worst performer. Recall that in Castalia* interferences are dynamically calculated. Our guess was that this calculation is somehow unaccurate, and thus we started a preliminary investigation to support our intuition, the results of which are shown in figure 7. The constant lines in the figure represent the average received packets of the 12n1g scenario as plotted in figure 5. The bold line represents the number of received packets in Castalia* when the dynamic calculation of the interferences is inhibited, and the interferences assume values between -200dBm and -80dBm. As can be observed in the figure, a value of about -100dBm seems to well fit the real data, while the dynamic calculation of interferences produces results similar to those obtained with a fixed value of about -92dBm, thus significantly under-estimating the number of received packets.
Concluding, Castalia is reliable, it is well designed and allows developers to easily implement their own functionalities and protocols and thus it is a good candidate as the reference simulation tool for WSNs. Further work is required to better understand the reason of the poor behavior of Castalia* in the considered scenario, but our preliminary investigation shows that the problem lies in the dynamic calculation of interferences.

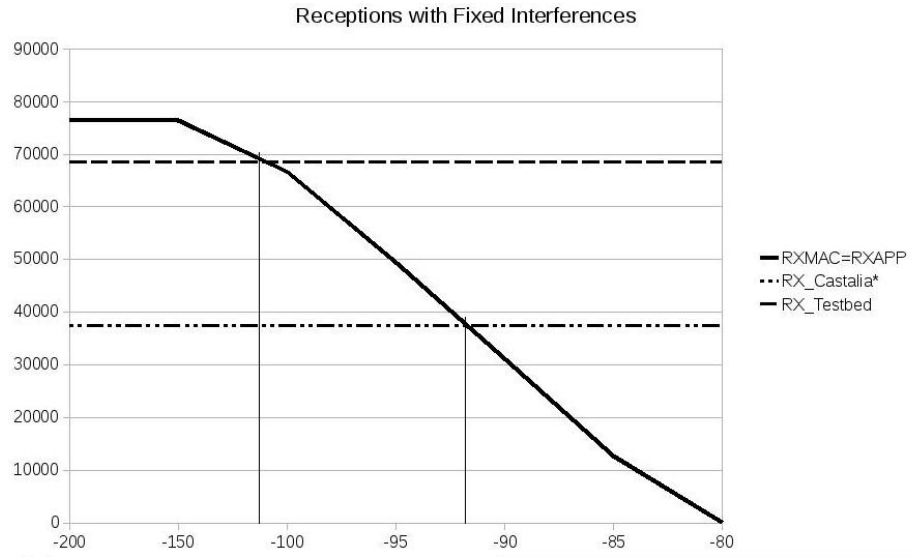

Figure 7: Behavior of Castalia* with fixed interferences(in dBm on X axis)

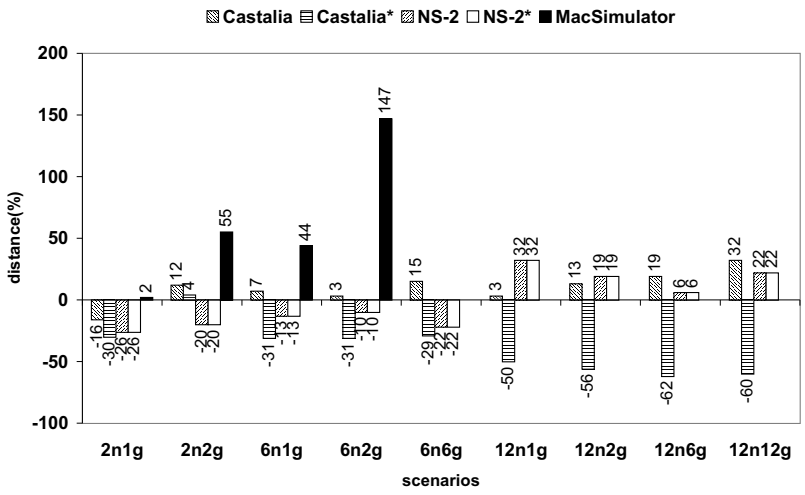

Figure 8: Evaluation of the X_{Sim}^{Scen} function on several scenarios

References

[1] http://www.isi.edu/nsnam/ns/.

[2] David Curren. A survey of simulation in sensor networks. http://www.cs.binghamton.edu/ kang/teaching/cs580s/ david.pdf.

[3] Dimosthenis Pediaditakis Hai N. Pham and Athanassios Boulis. From simulations to real deployment in wsns and back. *t2pWSN'2007*, 2007.

[4] David Kotz, Calvin Newport, Robert S. Gray, Jason Liu, Yougu Yuan, and Chip Elliott. Experimental evaluation of wireless simulation assumptions. In *MSWiM '04: Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pages 78–82, New York, NY, USA, 2004. ACM Press.

[5] H Nikookar and H Hashemi. Statistical modeling of signal amplitude fading of indoor radio propagation channels. volume 1, pages 84–86. 2nd International Conference on Universal Personal Communications, 1993.

[6] Sundeep Pattem, Bhaskar Krishnamachari, and Ramesh Govindan. Impact of spatial correlation on routing with compression in wireless sensor networks. April 26-27 2004.

[7] Theodore S Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 1996.

[8] A. Savvides S. Park and M. B. Srivastava. Sensorsim: A simulation framework for sensor networks. In *Proceedings of MSWiM 2000*, August 11, Boston 2000.

[9] Colesanti U.M., Crociani C., and Vitaletti A. On the accuracy of omnet++ in the wireless sensor networks domain: Simulation vs. testbed. *ACM*, 2007.

[10] Marco Zuniga and Bhaskar Krishnamachari. Analyzing the transitional region in low power wireless link. 2003.