MannaSim: A NS-2 extension to simulate Wireless Sensor Network

Rodolfo Miranda Pereira, Linnyer Beatrys Ruiz
Department of Informatics, State University of Maringa
Maringa, Parana, Brazil
e-mail: rodolfomp123@gmail.com, linnyer@gmail.com

Maria Luisa Amarante Ghizoni
Engineer School, Federal University of Minas Gerais
Belo Horizonte, Minas Gerais, Brazil
e-mail: malu.ghi@gmail.com

***Abstract*—A Wireless Sensor Network is a special kind of ad hoc network characterized by a large amount of nodes distributed over a real world environment, in order to monitor and send relevant data to an access point. In this paper, we present a framework for wireless sensors network simulation that was implemented as an extension for Network Simulator 2. This framework provides a way to configure the environment of the simulation by choosing parameters like the topology of the network, kind of data dissemination, type of routing protocol, hardware capacity and initial energy power of the sensor nodes. In order to demonstrate MannaSim's results, we also present a wireless sensor network scenario in which the proposed tool was used to simulate. Through the results provided by MannaSim, it was possible to choose the best configuration for the proposed scenario.**

Keywords–Wireless Sensor Network; Network Simulator (NS-2); Simulation Framework.

I. INTRODUCTION

Wireless Sensors Networks (WSN) are a class of distributed systems that recently is being target of a lot of research [1]. In a WSN, these data are captured by the sensor nodes in the environment that they are involved. Despite their variety, all WSNs have certain fundamental features in common. The essential is that they are embedded in the "real world". A WSN basically consists of sensor nodes deployed over a geographical area for monitoring physical phenomena like temperature, sound, vibrations, humidity, seismic events, chemical elements, and others. For this reason, WSN have many applications like smart environments in homes [2], buildings [3], transportation [4], disaster prevention [5] and even in the pharmaceutical field [6]. WSN have captured the attention and imagination of many researchers, encompassing a broad spectrum of ideas.

In the field of computer networks, in general, the simulation allows the evaluation of scenarios with a small cost and time compared to experiments in physical environments. This avoids unnecessary costs assembling real networks, and allows desired comparisons, improving decision-making power in the choice of the network parameters. In the case of WSNs, the simulation is even more advantageous, because it can be composed by a lot of sensor nodes, increasing the costs of this kind of network. Beyond this, different applications requires different types of sensor nodes, so the simulations can assist in decisions about the most appropriate sensor node to be utilized. Finally, changing and testing the configurations of the WSN certainly can help to understand the relation between the parameters.

Although simulation can bring many benefits, the choice of the wrong simulator may not result in satisfactory results. Simulations are very dependent of the functional model developed, and the model should optimally represent the environment simulated. MannaSim was made as an extension for the consecrated Network Simulator (NS-2) [7]. The framework consists of a set of base classes for the simulation of WSN, which may be specialized by users, and these classes extend the core of the NS-2 Simulator. MannaSim was built on existing model for WSN [8] and it is configurable in terms of sensing platform. As NS-2 Simulator, MannaSim is open source, thus the user can adapt the code if necessary.

MannaSim allows the user to configure detailed scenarios for simulations. Setting compositional requirements of the network (number of nodes, node type, density, dissemination type) and its organization (flat or hierarchical) MannaSim can accurately model different sensor nodes and applications while providing a versatile testbed for algorithms and protocols. After the simulation, MannaSim generate results, like the power level remaining in the components, the number and type of errors in the simulation, the average events division and the operation of the network in its lifetime. This kind of information can be analyzed and taken into account choosing the best configuration for the proposed WSN.

The rest of this paper is organized as follows: Section II presents some related works. The MannaSim extension is, as well as its design and the Script Generator Tool, presented in Section III. Section IV shows a simulation example, with the scenario description and the results of the simulations. Finally, in Section V, the conclusions and future works are presented.

II. RELATED WORK

A large number of WSN simulators has been published with different simulation outbreaks, requiring that developers are aware of these simulators so they can make the best choice of simulation in their projects. In the following, a review of some simulators already published will be shown.

Sensor Network Simulator and Emulator (SENSE) [9] is a simulation tool that is based on a component-oriented methodology that intends to promote extensibility and reusability to the maximum degree. It was designed in order to attend 3 kinds of users: high-level users, network builders and component designers. SENSE also proposes to be as efficient as NS-2 and as scalable as possible in its simulation. However, compared to the MannaSim framework, SENSE still lacks a comprehensive set of models and a wide variety of configuration templates that are required for wireless sensor network simulations.

SensorSim [10], which also has been built on top of NS-2, is a simulation framework for sensor networks. It provides sensor channel models, energy consumers, lightweight protocol stacks for wireless micro sensors, scenario generation and hybrid simulation. The sensor channel models the dynamic interaction between the sensor nodes and the physical environment.

 ISBN: 978-1-61208-398-8

At each node, energy consumers are said to control the power in multiple modes to efficiently use the power and prolong the nodes lifetime. When compared, MannaSim has much more scenarios configurations than SensorSim. Besides, SensorSim is no longer developed, therefore, no more available.

Another WSN simulator that we can quote is Avrora [11]. This framework is a cycle-accurate instruction level WSN simulator, that uses an event-queue model that allows improved interpreter performance and enables an essential sleep optimization. Its main goal is to provide the user the conditions to validate time-dependent properties of a large-scale WSN. Although Avrora intents to simulate the sensor nodes behavior at instruction level, it does not deal with the fact that nodes may run at slightly different clock frequencies over time due to manufacturing tolerances, temperature and battery performance. Compared to Avrora, MannaSim has a different purpose. While Avrora intends to provide a instruction level simulation, MannaSim proposes to give the user an event-based overview of the nodes communication.

Castalia [12] is a WSN simulator built on top of OMNeT++. Castalia features an accurate channel/radio model detailing radio behaviour. It also features a flexible physical process model, taking into account issues such as clock drift, sensor bias, sensor energy consumption, CPU energy consumption, and monitors resources such as memory usage and CPU time. Castalia's goal is to provide the user conditions to test algorithms and protocols in a wireless channel and radio model, with a realistic node behaviour relating to access of the radio. Thus, while Castalia focuses on simulating a radio model behaviour, MannaSim is aimed to provide a validation of the impacts of different settings and compositions on the errors and energy consumption of the sensor nodes.

NetTopo [13] is a framework for WSN simulation that has a visualization function to assist the investigation of algorithms in WSNs. NetTopo provides a common virtual WSN for the purpose of interaction between sensor devices and simulated virtual nodes. It allows users to define a large number of initial parameters for sensor nodes like residential energy, transmission bandwidth and radio radius. Users can also define and extend the internal processing behavior of sensor nodes like energy consumption and bandwidth management. For the visualization module, it works as a plug-in component to visualize testbed's connection status, topology and sensed data. Compared to MannaSim, NetTopo's goal is also different. NetTopo intends to assist the investigation of different algorithms impacts in a WSN, while MannaSim's goal is to investigate different compositional requirements impacts, like the network organization and number of nodes.

Table I shows a comparison between the main objectives and characteristics of MannaSim and others WSN simulation tools. In summary, it can be said that the main objective of MannaSim is to provide the user a first-order simulation for generic sensor nodes platforms in order to arrange an investigation of different network organizational and characteristics.

TABLE I. MANNASIM VS. OTHER SIMULATION TOOLS.

Framework	Characteristic	MannaSim
SENSE	Was designed in order to attend 3 kinds of users: high-level users, network builders and component designers, but still lacks a comprehensive set of models to configurate.	Was developed to first order validation users and has a lot of configurable parameters for the network composition.
SensorSim	Also has been built on top of NS-2, but is no longer developed.	Can give much more configurable parameters in the WSN, and is still under development.
Avrora	Simulate sensor nodes behavior at instruction level.	Gives the user a based on events overview of the nodes communication during network lifetime.
NetTopo	Its target is to investigate the impact of different algorithms in a WSN.	Its goal is to investigate the impacts of different network compositional requirements, such as organization and the number of nodes.
Castalia	Provides the user conditions to test algorithms and protocols in a realistic model of the nodes communication.	Proposes a first order validation in the chosen composition of the WSN.

III. THE MANNASIM FRAMEWORK

MannaSim is composed of two solutions: The Framework and the Script Generator Tool (SGT). The Framework is a module for WSN simulation based on the Network Simulator (NS-2) and SGT is a front-end for the creation of simulation scripts. MannaSim's home page [14] gives detailed informations about scenario configuration and also presents scenario examples. It is important to note that MannaSim is being developed under the GNU General Public License, in other words, it intends to guarantee the user freedom to share and change all versions of the code. In the following subsections, we present details about the framework.

A. On the MannaSim Design

MannaSim inherits the core features of the Network Simulator (NS-2), and builds up new features that include ability to use different protocol profiles for different WSN applications, different sensor parameters and distribution. The requirements of the network composition, like number of nodes, types of nodes, density, flat or hierarchical organization are different for each application. Thus, a WSN Simulator has to be flexible enough to attend this kinds of characteristics. The goal of MannaSim is to be flexible to make a detailed WSN simulation which can accurately model different sensor nodes and applications.

The first step taken in the implementation of the simulator was the implementation of a node specific to WSNs, the sensor node. Since NS-2 already possesses an object class that represents a mobile node with wireless communication capability, the new node was implemented extending the mobile nodes class. To this new node, new characteristics were added such as sensing and processing energy consumption, 'wake up' and 'sleep' functions and control of components usage state such as sensor devices and processor. A subclass of the existing energy model was also created; it implements a battery class that can be used to implement the different existing battery models. Next, specialized classes that describe the behavior of each node type found in a WSN were modeled and implemented. These behaviors were implemented in the application layer, since no restriction may be imposed to the user regarding the desired protocol stack. Thus, each developed class that models a node from MannaSim inherits from NS's application. Common-nodes, leader nodes and access points

 ISBN: 978-1-61208-398-8

were created also. Figure 1 shows the simplified class diagram of MannaSim.

Following the characteristics of a WSN, below we present how MannaSim's classes were designed to attend the possible features in a WSN.

- **Simulate different kinds of sensor devices:** Data collection in MannaSim is simulated through the generation of artificial informations. The class DataGenerator is the basis for the generation of information. Simply by extending it the user can simulate a variety of sensors devices. The artificial data collected must be encapsulated in a corresponding class. This class represents the data that will be disseminated by the sensor nodes in the WSN towards the AP.
- **Contemplate different sensing options:** MannaSim allows the continuous collection, periodically and on demand. The frequency which the data are generated by the inherited classes of DataGenerator models the different types of sensing. For networks that use scheduled or continuous collection, a timer (SensingTimer) is used. The demand network only performs the collection when a requisition is requested by the observer.
- **Contemplate different disseminating options:** MannaSim allows continuous, scheduled or on demand data dissemination, regardless of the chosen sensing type. For example, the network can collect data continuously, but spreads them periodically. For networks that utilize programmed dissemination, a timer (DisseminatingTimer class) is used. The demand network performs dissemination only if the observer sends a request. The requests are modeled by the class OnDemandData. Each request can contain multiple queries, which are instances of the class OnDemandParameter, it specifies the data of interest to the observer. WSNs with continued dissemination transmit data as soon as they are collected and processed. Messages of data that are sent to head nodes or to the AP, are modeled by the class SensedData, which is an implementation of the abstract class AppData API2 the standard NS-2.
- **Contemplate different processing options:** In MannaSim, all data collected pass through some kind of processing before being disseminated. The base class Processing serves as a starting point for the creation of specific types of processing for each application. The Processing of requests on demand are implemented in this class.
- **Allows the simulation of flat and hierarchical sensor networks:** The behavior of the sensors nodes was implemented in MannaSim as a protocol from the application layer in NS-2, using the inheritance from the Application class. The general behavior of a sensors is implemented in the class sensorBaseApp. This class have the basis for creating different types of behavior such as, for example, the head (class ClusterHeadApp) or the common (class CommonNodeApp). The creation of different behaviors allows modeling of hierarchical multilevel WSNs.
- **Allows the simulation of homogeneous and heterogeneous sensor networks:** Through the SensorNode class, inherited of the MobileNode class from NS-2, the MannaSim is able to create sensors nodes with different settings. Each sensor node has an object Battery class, a specialization class from EnergyModel of NS-2. This class defines a battery model for the sensors. If extended, new models of energy decay can be created in the simulations.
- **Simulate networks with one or more Access Points:** The MannaSim allows that a simulation of a WSN has one or more APs. The class AccessPointApp enables the communication of the network with the external observer. One or more nodes of the sensor network, or even a node that is not a sensor node, may contain this application and act as an AP.
- **Allows the utilization of different protocols:** Different routing protocols (specific or not to WSNs, single or multi-hop) can be used. The same applies to the transport and link layers. The use of protocols in these layers, in the simulations with the MannaSim, follows the same procedure for any NS-2 simulation. Two of the most popular routing protocols for WSNs (LEACH [15] and Directed Diffusion [16]) are already implemented in MannaSim.

B. The SGT and the MannaSim's Settings

The simulation in NS-2 involves creating scripts in TCL scripting language. This is a tedious and error-prone task, since there are several parameters that need to be adjusted. In order to simplify this task, it was developed an automated system for the generation of TCL scripts used by MannaSim, which is the SGT. Through a friendly interface, the user specifies values for the main parameters of the network. Then the tool takes care of creating the corresponding TCL scripts. The Script Generator Tool is composed by 4 different user interfaces:

Basic Configuration: In this interface we can configure the basic settings of the wireless sensor network simulation like: Transport Protocol (TCP or UDP); Routing Protocol (DSR, TORA, LEACH, Directed Diffusion, DSDV or AODV); MAC (Only IEEE 802.11 is available); Link Layer (Only NS-2 LL default link layer is available); Antenna; Radio Propagation (FreeSpace, Shadowing, ShadowingVis or TwoRayGround); Interface Queue (DropTail, DropTail/XCP, RED, RED/Pushback, RED/RIO, Vq or XCP); Interface Queue Length; Scenario Size; and Simulation Time.

Access Point: Through this interface we can set the AP configurations such as: Number, location, Initial Energy and Transmission Range of the Access Points.

Cluster Head: This interface can be used to configure the Cluster Head settings like: Number, location, Initial Energy and Transmission Range of the Cluster Heads; Transmission Range; Processing Type (Only Aggregate Processing is available); Dissemination Type (Continuous, On Demand or Scheduled); and Dissemination Interval.

Common Node: This interface can be used to set the Common Nodes (CN) settings. Beyond the same parameters that can be configured for Cluster Heads, the Common Nodes can also receive parameters like: Sensing Type (Continuous, On Demand or Scheduled); Sensing Interval; Data Generator Type (Only Temperature and Carbon Monoxide are available, but others can be implemented); Data Average Value; Data Standard Deviation; and Maximum Data Value.

 ISBN: 978-1-61208-398-8

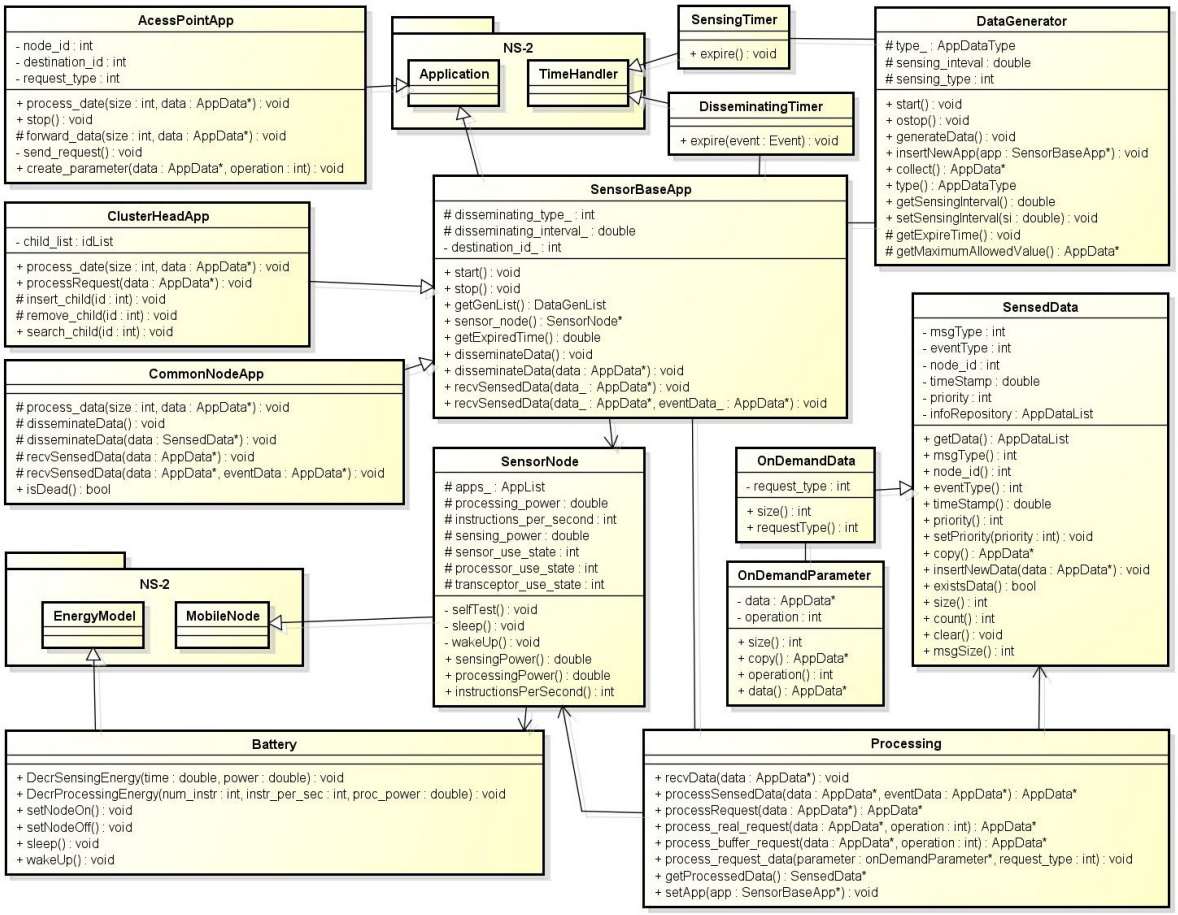

Figure 1. MannaSim's simplified class diagram.

IV. USING MANNASIM FRAMEWORK

To create simulation scenarios using MannaSim, the user must set up the desired parameters of the sensor nodes, and then create node instances and their applications using OTcl language [17]. There are no restrictions regarding the scenarios configuration that may present different compositions, organizations, hierarchical levels, number of nodes, number of access points, and so on.

In order to show how MannaSim works and what kind of results are provided, the following subsections present a particular scenario description and the results provided by MannaSim.

A. The Scenario Description

The particular scenario proposed is a smart home called Follow-Us [2]. This is an application for elderly people monitoring that considers the home instrumentation with WSN technology, clothes manufacturing with wearable computing technology, and the application of concepts involving Social Sensing and Internet of Things. The proposed environment organizes the collected data flow from different types of sensors and networks, establishes a processing routine according to application objectives and provides commands that allow forward the information to be used as control parameters of the environment itself or used by external agents.

Two types of simulation scenarios were developed. The first scenario assumes that the environment is being sensed room by room, and the sensors are on standby while the elderly is in another room. For this scenario, a hierarchical organization in the sensor nodes was used. The second scenario considers that the whole house is being sensed disregarding the rooms divisions, in other words, all the sensors are active all the time. For the second scenario, a flat organization was used in the sensor nodes.

TABLE II. THE SIMULATIONS PARAMETERS

Transport Protocol	TCP
Routing Protocol	AODV
Node Type	MicaZ
Dissemination Type	Scheduled
Dissemination Interval	5 seconds
Initial Energy for Common Nodes	30 Joules
Initial Energy for Access Points	100 Joules
Antenna Range of Common Nodes	10 meters
Antenna Range of Access Point	30 meters
Number of Common Nodes	30
Number of Access Points	3
Scenario Area	30 meters^2

Table II presents the principal parameters used in the simulations. Figure 2 shows how the CNs and the APs were spread over the house in the simulation. Each scenario was executed 33 times, to reach a convergence of the result data, and the simulation time was set to 150 units, time enough to guarantee that the models have been warmed-up sufficiently so that the samples collected will have statistical validity.

 ISBN: 978-1-61208-398-8

Figure 2. The deployment of the sensor nodes and access points over the house scenario.

B. The Results Provided by MannaSim

While the flat organization gives only one group of network for the entire house, the hierarchical organization has to be simulated as eight groups of subnetworks, one for each room of the house that is named in Figure 2. This differentiation is due the fact that in the hierarchical organization each subnetwork is independent and can be inactive for a while. MannaSim provided several results such as described below. It is important to note that only the data shown in this Section was provided by MannaSim's results, the graphs shown in the figures were generated using other tools.

Power Level: Figure 3 shows the power level remaining in the components in each room (for the hierarchical network organization), while Figure 4 presents the power level remaining in the components in each simulation (for the flat network organization).

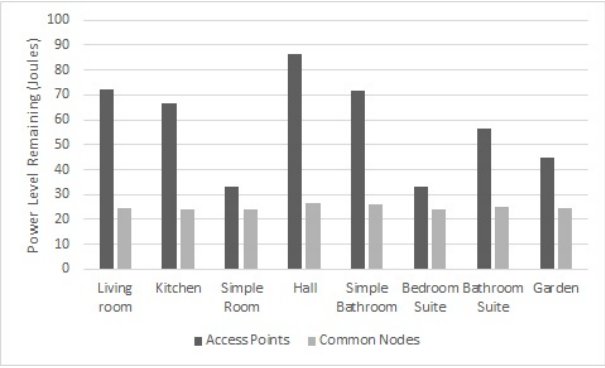

Figure 3. The average power level remaining in the components per room.

Error Types: The average number of errors division in both types of network organization, shown in Table III and Table IV. In this tables the errors are divided by Uninformed, Excessive number of Address Resolution Protocol packets (ARP), The MAC layer was not able to transmit the packet

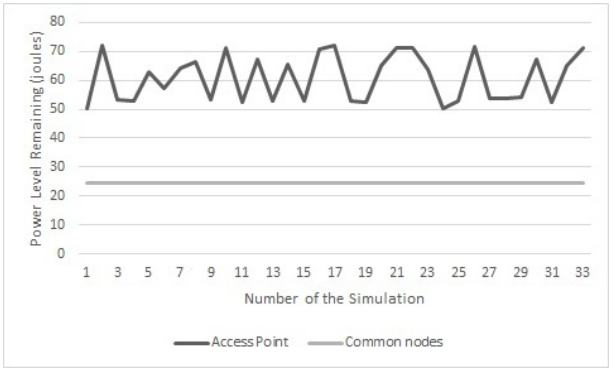

Figure 4. The average power level remaining in the components per simulation.

(CBK), Packet collisions (COL), Excessive packets in the interface queue (IFQ) and Sending packets excessively (RET).

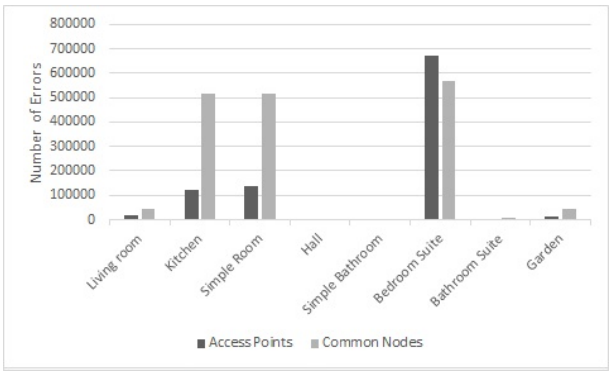

Figure 5. The average errors in the components per room.

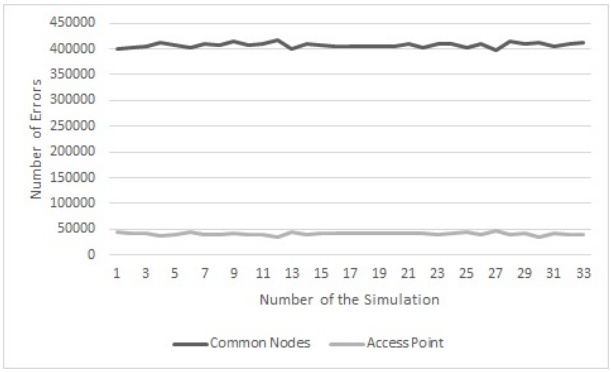

Figure 6. The average errors in the components per simulation.

Number of Errors: The average number of errors in the components in each room (for the hierarchical network organization), presented in Figure 5, or in each simulation (for the flat network organization), presented in Figure 6.

Simulation Events Division: The average events division (Error, Routing, Receiving, Transmitting) in both types of network organization, presented in Figure 7.

Based on the results, it is possible to analyze the WSN from different points of view. In the hierarchical network simulation the average power consumption of the sensor nodes is 5 Joules whereas for the access points is 40 Joules. Although, in the flat network simulation, the average power consumption of the sensor nodes is 5.6 Joules and for the access points is 39

TABLE III. THE AVERAGE ERRORS EVENTS DIVISION IN THE HIERARCHICAL NETWORK ORGANIZATION

Errors Types	Common Nodes	Access Points
Uninformed	891047	0
ARP	11	10
CBK	92	0
COL	63077	1692069
IFQ	6763	0
RET	184	0
Total	961174	1692079

TABLE IV. THE AVERAGE ERRORS EVENTS DIVISION IN THE FLAT NETWORK ORGANIZATION

Errors Types	Common Nodes	Access Points
Uninformed	12619826	21210
ARP	2773	206
CBK	9800	146
COL	603044	1341910
IFQ	193115	0
RET	19000	298
Total	13447558	1363770

Joules. Taking into consideration these results, it is possible to conclude that the power consumption is almost the same in both scenarios.

The average number of errors in the flat network simulation is almost 6 times superior that the average number of errors events in the hierarchical network. Whereas the hierarchical network got a total of 2.653.253 errors, the flat network got an average of 14.811.328. In Table III, for the hierarchical network organization, we can see that packet collisions are the most frequent errors, followed by uninformed errors. Meanwhile, Table IV shows that, for the flat network organization, most part of the errors are uninformed and, after that, packet collisions are the most frequent errors.

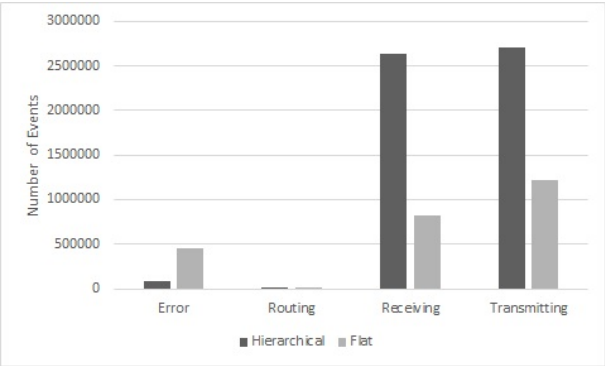

Figure 7. The average events division in the simulations

Figure 7 indicates that if we compare in proportion terms, error events were only 1% of the total events in the hierarchical network, while in the flat organization network, errors took approximately 18% of events. This information shows that, giving the chosen configurations, the hierarchical network would bring a smaller error percentace than flat network, therefore, it would be more appropriate to use in the simulated application proposed.

The impacts of different network organizations over the total errors is a kind of information that cannot be observed by other simulation tools, such as the ones presented in related works, even SensorSim, which also extends NS-2 core. The principal disadvantage of MannaSim is that the power consumption module is under development, so it does not considers factors such as the executed code in the sensor nodes to make its estimatives yet.

V. CONCLUSIONS AND FUTURE WORKS

Select the correct level of detail (or level of abstraction) for a simulation is a difficult task. Few details can produce simulations that are misleading or incorrect. On the other hand, add to many details requires more time to implementation and debugging, which probably will slow down the simulation and distract from the research problem. In this meaning, it is relevant that there is a simulation tool that helps the WSNs developers to build their scenarios, as well as, performs their experiments in order to obtain results that permits proves their theories. The MannaSim framework allows that different WSN scenarios can be simulated, offering possibilities for the configuration of WSNs and applications. Furthermore, it provides a set of base classes that can be extended, making the framework specific to the needs of researchers who use it.

The NS-3 tool, a later version of NS-2, whose core is used by MannaSim, was published in [18]. The migration of MannaSim's core to NS-3 is a future work point. However, as the NS-3 project was started "from the beginning", this migration is subject to a study of the changes needed in MannaSim's set of classes so that the interface with NS-3 can be made. Furthermore, the migration should not be made while there are no scientific studies that prove the efficiency and effectiveness of the NS-3.

As future works in MannaSim, it can also be cited the construction of a realistic battery energy decay model, considering the influence of ambience temperature in the battery capacity. To improve the power of observation of the simulation results, we intend to create a graphical interface for the MannaSim's output. Beyond this, other features, like comunication interference between the nodes, will be implemented. Finally, a more ambitious project intends to add to MannaSim the feature of Instruction Level simulation.

ACKNOWLEDGMENT

We would like to thank Thais Regina de Moura Braga and Fabricio Aguiar Silva for their commitment to the development of MannaSim Framework. We also thank the National Council for Scientific and Technological Development (CNPq) from the brazilian government for it is financial support on the project (Project process number 55.2111/2002-3).

REFERENCES

[1] V. Gungora, B. Lu, and G. Hancke, "Opportunities and challenges of wireless sensor networks in smart grid," IEEE Transactions on Industrial Electronics, vol. 57, no. 10, October 2010, pp. 3557–3564.

[2] M. L. A. Ghizoni, A. Santos, and L. B. Ruiz, "Follow-us: A distributed ubiquitous healthcare system simulated by mannasim," Computational Science and Its Applications, vol. 7336, June 2012, pp. 588–601.

[3] T. Torfs, T. Sterken, S. Brebels, and J. Santana, "Low power wireless sensor network for building monitoring," IEEE Sensors Journal, vol. 13, no. 3, March 2013, pp. 909–915.

[4] R. A. Khan, S. A. Shah, and M. A. Aleem, "Wireless sensor networks: A solution for smart transportation," Journal of Emerging Trends in Computing and Information Sciences, vol. 3, no. 4, April 2012, pp. 566–571.

 ISBN: 978-1-61208-398-8

[5] G. N. L. R. Teja1, V. K. R. Harish, N. M. Khan, R. B. Krishna, R. Singh, and S. Chaudhary, "Land slide detection and monitoring system using wireless sensor networks (wsn)," in IEEE International Advance Computing Conference, Gurgaon, February 2014.

[6] C. K. Ng, C. H. Wu, L. Wang, W. H. Ip, and J. Zhang, "An rfid-enabled wireless sensor network (wsn) monitoring system for biological and pharmaceutical products," in International Symposium on Computer, Consumer and Control, Taichung, June 2014.

[7] "The network simulator (ns-2) home page," http://www.isi.edu/nsnam/ns/, accessed on November, 2014.

[8] L. B. Ruiz, J. M. Nogueira, and A. F. Loureiro, "Manna: A management architecture for wireless sensor networks," IEEE Communications Magazine, vol. 41, no. 2, February 2003, pp. 116–125.

[9] G. Chen, J. Branch, M. Pflug, L. Zhu, and B. Szymanski, "Sense: A wireless sensor network simulator," in Advances in Pervasive Computing and Networking, 2005, pp. 249–267.

[10] S. Park, A. Savvides, and M. B. Srivastava, "Simulating networks of wireless sensors," in Proceedings of the Winter Simulation Conference, vol. 2, Arlington, December 2001, pp. 1330–1338.

[11] B. Titzer, D. Lee, and J. Palsberg, "Avrora: Scalable sensor network simulation with precise timin," in Proceedings of the 4th International Symposium on Information Processing in Sensor Networks, April 2005, pp. 477–482.

[12] A. Boulis, "Castalia: Revealing pitfalls in designing distributed algorithms in wsn," in Proceedings of the 5th International Conference on Embedded Networked Sensor Systems, 2007, pp. 407–408.

[13] L. Shu, M. Hauswirth, H.-C. Chao, M. Chen, and Y. Zhang, "Nettopo: A framework of simulation and visualization for wireless sensor networks," Ad Hoc Networks, September 2010, pp. 799–820.

[14] "The mannasim's home page," http://www.mannasim.dcc.ufmg.br/, accessed on December, 2014.

[15] B. Wang, C. Shen, and J. Li, "Study and improvement on leach protocol in wsns," Automatic Control and Artificial Intelligence, March 2012, pp. 1941–1943.

[16] N. El-Bendary, O. Soliman, N. Ghali, A. E. Hassanien, V. Palade, and H. Liu, "A secure directed diffusion routing protocol for wireless sensor networks," Next Generation Information Technology, June 2011, pp. 149–152.

[17] "Otcl home page," http://otcl-tclcl.sourceforge.net/otcl/, accessed on April, 2015.

[18] T. R. Henderson, M. Lacage, and G. F. Riley, "Network simulations with the ns-3 simulator," in Proceedings of the Special Interest Group on Data Communication Conference, Seatle, Washington, August 2008.

 ISBN: 978-1-61208-398-8