Simulation of Protocol Stacks for Internet of Things

João Paulo Andrade Lima
Federal University of Sergipe
Av. Marechal Rondon, s/n Jardim Rosa Elze
+55 (79) 2105-6600
dm.joaopaulo@gmail.com

Diego Assis Siqueira Gois
Federal University of Sergipe
Av. Marechal Rondon, s/n Jardim Rosa Elze
+55 (79) 2105-6600
diego.se.ita@gmail.com

Admilson de Ribamar Lima Ribeiro
Federal University of Sergipe
Av. Marechal Rondon, s/n Jardim Rosa Elze
+55 (79) 2105-6600
admilson@ufs.br

ABSTRACT
Looking across this growing on the Internet of Things scenario, the fact that it is an expansion area, recently, poorly studied and documented; and the need to have results as a basis for study and future work This work shows the simulation of the communication stack uIP using the IPv6 protocol and the Rime communication stack in the context of IoT, using the COOJA simulator, which is included in Contiki system. As well aims to publish the results in the academy aiming to provide a didactic basis for the study and future research in the area in question.

Categories and Subject Descriptors
I.6.0 [**Simulation and Modeling**]: General.

General Terms
Documentation, Performance, Measurement.

Keywords
Internet of Things; Contiki; COOJA; discrete simulation.

1. INTRODUCTION
Imagine the existence of a network that would make all our devices cooperate in every moment, to talk spontaneously among themselves and with the rest of the world, and together compose a kind of virtual single computer - the sum of their intelligence and knowledge [7]. This technological advancement in wireless scenario has reached the physical world with the integration of network sensors and the incorporation of communication technology in everyday life objects. This idea of ascension of the Internet was invented by the founder of the MIT Auto-ID Center, Kevin Ashton in 1992 [1] and was called the Internet of Things (IoT).

However, the IoT is characterized by the insertion of daily objects on the Internet, and the problems relating to this are the limited storage capacity, processing and power of such objects. As a possible solution, Adam Dunkels presented [2] an implementation of the communication stack TCP/IP for integrated ultra-low power processing devices. For this feat, the implementation of uIP (as it became known such stack) is designed to have only the minimum set of features required for the operation of the TCP/IP stack.

Alternatively, Adam Dunkels also implemented the layered Rime communication stack [3], which simplifies the implementation and complexity of communication protocols. The abstraction layers of this stack are extremely simple both in terms of interface and implementation, and are combined to form powerful high-level abstractions.

Analyzing across this growing scenario about the Internet of Things, the fact that it is an expanding area, recently, little studied and documented. And knowing the work [2] that presents the operating system Contiki and the knowledge about COOJA simulator, included in this, which allows the simulation of these motes IoT both at the network level, at the operating system level, and the set of the machine instruction level; and has the practicality of allowing the use of the code implemented in simulator to the physical mote without changing.

Besides the world of [6] where the authors performed several simulations of computer networks using a special simulator, and published the results in the academy by serving as a basis for future study. This paper performs the simulation of uIP and Rime communication stack using the IPv6 protocol and COOJA simulator, aiming to publish the results in the academy to provide the didactic baseline study and future research in the area in question.

TCP, UDP, REST and RIME stack (via ABC module) protocols were simulated plus a small HELLO WORLD to demonstrate the operation of the simulator and the Contiki operating system running, all these simulations were performed with the use of IPv6. This paper will detail only the UDP protocol due to the limit imposed on the paper size.

This work was divided into several chapters, the first chapter is introduction. The next chapter is related work in the area studied. The chapter three exposes the simulation planning and the chapter four presents the details of the UDP simulation which is the focus of this work.

Chapter five presents the results and holds a discussion about them. Already the chapter six shows the conclusion and proposed future work.

2. RELATED WORK
A relevant work in this area is the Michael Kirsche work [4], where the author proposes a hybrid simulation environment that aims to perform Internet of Things studies. The main goal is to simulate correctly the mote in system-level and in network-level at

the same time providing a simulation that lets you test your application protocols during pre-deployment, given that the COOJA [5] and OMNeT++ [9] simulators have strengths in their respective areas, but do not cover the mote in the system-level and network-leave at the same time, the author proposed this environment to reduce the gap between research and practice development.

Otherwise, the work [5] is also related to the central theme of this paper, since it presents the COOJA simulator to simulate a wireless sensor network. This simulator was chosen for this work, mainly due to its cross-level simulation, enabling the simultaneous simulation at various levels of the system.

Knowing that RPL (IPv6 Routing Protocol for Low Power and Lossy Networks) is the IETF standard candidate for IPv6 routing in low-power wireless sensor network, Tsiftes, Eriksson and Dunkels presents the first results that were obtained with the ContikiRPL implementation [8]. The COntikiRPL is an implementation of the RPL routing protocol for low power and lossy networks. However, according to the authors, the studies about RPL proved that practical experience of RPL implementations on systems with limited resources, similar to the IoT objects, has been lacking. Therefore, the authors have developed the RPL within the µIP stack, running experiments both in a low power wireless network as in the simulation.

As can be seen, no work to simulate completely the µIP stack with the IPv6 protocol was found, thus showing the necessity of simulation and publishing the results in the academy to serve a didactic basis for the study and development of new applications.

3. SIMULATION PLANNING

As main materials used were the Contiki operating system with the simulator COOJA, available in the same, the simulations were performed in the virtual mote TmoteSky offered by COOJA.

The first step was to make the assembly the Contiki environment. Once the environment is ready to work, a survey of key metrics for the simulation of IoT motes was done, they are:

- Number of sent packets;
- Number of received packets;
- Number of dropped packets;
- Number of retransmitted packets;
- RSSI – Indicator of the intensity of the signal received;

And for the Rime communication stack, outside the above mentioned:

- The CPU consumption;
- The consumption used by each LED in every moment of the simulation;
- The consumption to transmit and receive packets.

Besides the metrics, was chosen a protocol for each communication stack to perform the simulations:

- The simple UDP, UDP, TCP and REST protocol for the uIP stack;
- The communication module ABC (Anonymous best-effort local area broadcast) to the Rime stack;

After that, the MAKE tool was used to assist in compiling the codes of the Contiki simulations.

4. SIMULATIONS

With the intention to compare the results, several simulations for each protocol with applying some changes were made in the scenarios, they are:

- Transmission success rate (TX);
- Receipt success rate (RX);
- Motes arrangement (random or arranged manually).

The transmission and receipt success rate specifies the package chance being delivered or received by a particular mote, this rate is of fundamental importance in protocols simulations that need to establish a connection like the TCP and REST, because there are not handshake if the sent packets not arrive in the other particular mote. As the TX and RX, the arrangement of the motes is also critical in simulations, because if overcome certain limit, set to 100 meters, is not possible for communication.

Both TCP and REST protocol were simulated using a client-server system, where they were placed clients and one server trying to respond to all customers requests, these requests were not always answered successfully, either by change the transmission or receipt success rate (by a large distance between the motes), for having drop packets while trying to establish a connection form client to server or due the competition to multiple clients sending packets simultaneously to a single server, wich is unable to answer all clients simultaneously.

Already in the UDP and ABC module (Rime stack) protocols were made broadcasts in the network so that it was not necessary to answer requests a server, in this case all motes send packets to the other and all have listening on, so it is possible to remove metrics without the use server. In the simulation of these protocols the transmission and receipt success rate variation was less decisive than in the case of protocols that require a connection establishment, but the motes arrangement was as decisive as in TCP and REST.

For the UDP protocol, there are two distinct ways to simulate it, using the simple UDP API or UDP API. The simple UDP API is implemented by the Contiki and provides functions that facilitate the simulation of this protocol, bacause is not easy to work with UDP in Contiki. In this paper, simulations were performed with both API using simple UDP and UDP.

Even with a difficulty degree a bit higher, we will treat about the Contiki UDP API simulation. To perform this simulation first reviewed for the compilation rules necessary for the UDP implementation, these rules are:

- CONTIKI=/home/user/contiki
- Include $(CONTIKI)/Makefile.include
- WITH_UIP6=1: Command that enables the use of the uIP stack, in this case with IPv6.
- UIP_CONF_IPV6=1: It is a flag used to enable IPv6 because by default the Contiki uses IPv4.
- UIP_CONF_IPV6_CHECKS=1: It is also a flag, but he is responsible for ensuring that packets arriving to the mote are correctly formed.

- UIP_STATISTICS=1: It is responsible for capturing the metrics of incoming packets sent and lost.

With the Makefile file ready was sought to implement the execution code of the mote. Was first added to the core libraries Contiki: "contiki.h" and "contiki-net.h", then the standard output: <stdio.h> and lastly the "cc2420.h" which includes functions to capture RSSI floor values and last packet received RSSI.

As this simulation sends a broadcast to all motes using the UDP API, you should first use the udp_broadcast_new function to create a new connection UDP broadcast. Created the connection, only the uIP stack can choose when to send packets, however tcpip_poll_udp function forces the stack to capture the specified connection, thus allowing the sending of packets at the desired time.

In this simulation, after the stack choosing the connection, it is used the uip_send function to send a string via broadcast connection. After each sent, print up the metrics of uip_stat structure and presents the RSSI values with the functions:

- cc2420_rssi(): Reading RSSI floor;
- cc2420_last_rssi: RSSI of the last packet received.

5. RESULTS AND DISCUSSION

The simulation of UDP API observed the UDP packets in environments with fixed and random arrangement of motes in the environment with the random arrangement, the UDP packets of analyzed 10 motes are shown in Figure 1.

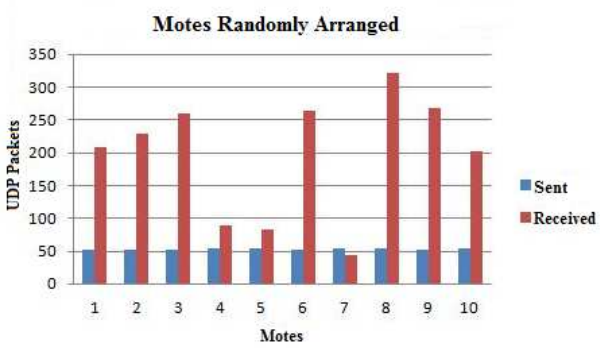

Figure 1. Histogram of UDP packets sent and received with motes scattered randomly by COOJA.

Note that this graph confirms the relationship of the randomness of the motes with the received packets, since all devices have linearity in the quantity of packets sent, but a variation in relation to those. This explains the fact of the motes 4, 5 and 7 have received fewer packets, since they were farther apart from the rest.

To try to confront these values, was built an environment with fixed arrangement, where all motes are in the transmission area of all others. With the environment ready was obtained the graph of Figure 2.

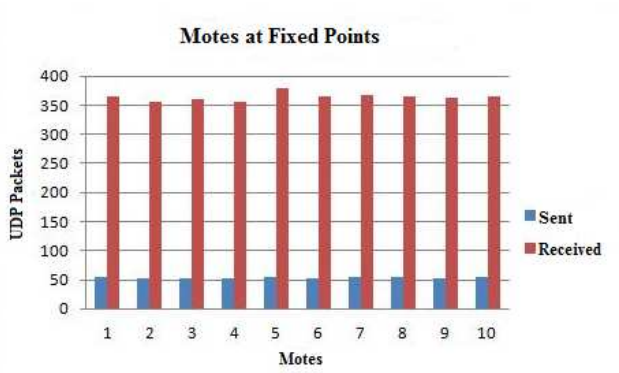

Figure 2. Histogram of UDP packets sent and received with motes at fixed points.

Using this plot can confirm the dependence of the arrangement of motes, realize that occurs a permanence of packets received and sent around a value, showing that all are participating equitably in this simulation of broadcast.

Another point that was discussed in this simulation was the relationship of the motes with TX and RX, Simulating them in scenarios with motes arranged randomly and arranged in fixed points, to observe the impact of changes in the rate of success. Following what was planned, was altered the TX and RX to 50% in the environment with the random arrangement and were obtained the values of the figure 3.

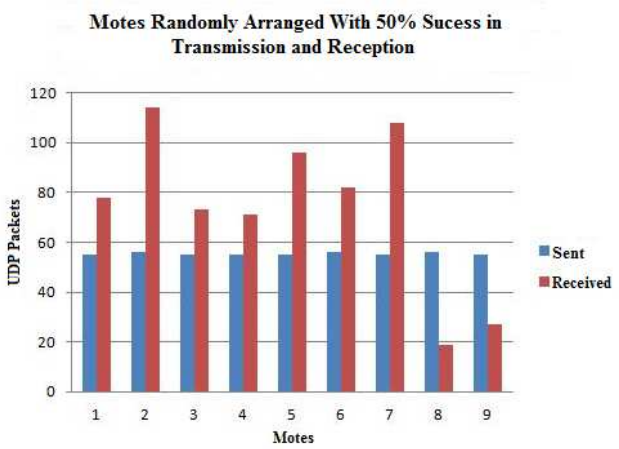

Figure 3. Histogram of UDP packets sent and received with motes scattered randomly by COOJA with TX and RX at 50%.

Note that occurs again a fixity in the amount of packets sent while arises an alternation in the received packets. The amount of received packets also decreases due to reduced success rate.

In the same line of thought, simulated an environment with fixed arrangement with a reduction of 50% success rate. The values obtained are shown in Figure 4.

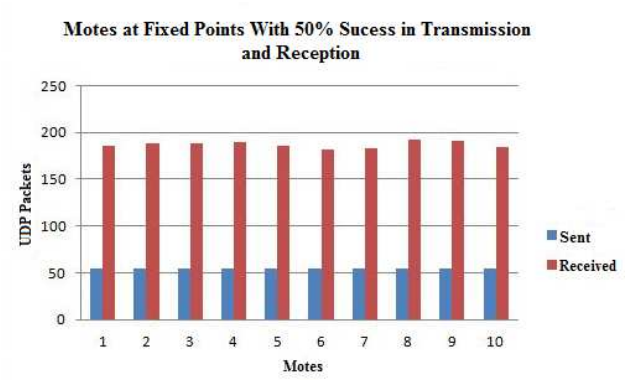

Figure 4. Histogram of UDP packets sent and received with motes at fixed points and TX and RX at 50%.

As was expected, the simulation obtained similar values to the histogram of Figure 2, but with the decrease in the amount of packets received due to reduction success rate.

6. CONCLUSION AND FUTURE WORK

The event-based programming required at the Contiki is not simple, in addition, many of the settings needed to run the simulation are not available in the documentation, for example, have the codes described in the makefile.

Another problem encountered was in relation to the Contiki libraries, which sometimes caused problems because their functions do not work as they should, as an example, we have the uip_statics function. After extensive analysis of the uIP.h library it was verified that there was an error in this, where #if UIP_STATISTICS = 1 is found, it is necessary to exchange for just #if UIP_STATISTICS and at the end of the definition just let the #endif.

Despite the difficulties encountered due to be an area that there is little material available for consultation and a rather poor documentation, satisfactory results were obtained in the simulations. These simulations show that the protocols also depend on the environment in which the motes are inserted.

The simulation performed in this paper aimed to show that it is possible to implement in real motes the Internet of Things. Thus, future work proposed in this paper, is the realization of the simulation of these protocols in real motes, observe their behavior and compare the results obtained by the simulation with the results obtained by real motes.

7. REFERENCES

[1] ASHTON, K. **That 'Internet of Things' Thing**. Online RFID Journal. Published in 2009. Available in: <http://www.itrco.jp/libraries/RFIDjournal-That%20Internet%20of%20Things%20Thing.pdf>. Acessado em 23 de setembro de 2013.

[2] DUNKELS, A. **Full TCP/IP for 8-bit architectures.** In: Proceedings of The First International Conference on Mobile Systems, Applications, and Services (MOBISYS 2003) .May 2003.

[3] DUNKELS, A. **Rime - a lightweight layered communication stack for sensor networks.** In: European Conference on Wireless Sensor Networks (EWSN). January 2007, Delft, The Netherlands.

[4] KIRSCHE, M. **Simulating the Internet of Things in a Hybrid Way**.Proceedings of the Networked Systems (NetSys) 2013 PhD Forum. Published in March 2013. Available in: <https://www-rnks.informatik.tu cottbus.de/content/unrestricted/staff/mk/Publications/NetSys _2013-PhD_ForumKirsche.pdf>.

[5] ÖSTERLIND, F.; DUNKELS, A.; ERIKSSON, J.; FINNE, N.; VOIGT, T. **Cross-Level Sensor Network Simulation with Cooja.** Proceedings of the First IEEE International Workshop on Practical Issues in Building Sensor Network Applications. (SenseApp 2006), Tampa, FL, USA, 14 November 2006.

[6] PETERSON, L.L., DAVIE, S. B. **Computer Networks: A System Approach.** 2003. Third Edition: A Systems Approach, 3rd Edition.

[7] SUNDMAEKER, H. **Vision and Challenges for Realising the Internet of Things**. 2010.

[8] TSIFTES, N.; ERIKSSON, J.; DUNKELS A. **Low-power wireless IPv6 routing with ContikiRPL.** Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks. April 12-16, 2010, Stockholm, Sweden [doi>10.1145/1791212.1791277].

[9] VARGA, A.; HORNING, R. **"An Overview of the OMNeT++ Simulation Environment".** Proceedings of the First Conference on Simulation Tools and Techniques for Communications, Networks and Systems. (Simutools 2008). ICST, 2008, PP. 1-10.