# Improving NS-2 Network Simulator to evaluate IEEE 802.15.4 wireless networks under error conditions

André Guerreiro, Jeferson L. R. Souza and José Rufino
*University of Lisbon - Faculty of Sciences*
*Large-Scale Informatics System Lab. (LaSIGE)*
*aguerreiro@lasige.di.fc.ul.pt, jsouza@lasige.di.fc.ul.pt, ruf@di.fc.ul.pt*

Keywords: NS-2 simulator, wireless communications, network inaccessibility, timeliness, dependability

Abstract: The behaviour of wireless networks in the presence of error conditions is still being studied by the research community. Improvements on the evaluation methods and tools are crucial to acquire a better knowledge, and understanding of the network operation under such conditions. This paper presents enhancements on the network simulator NS-2 to support the evaluation of the IEEE 802.15.4 standard, used as a case study. We are specially interested to evaluate the temporal behaviour of the network operation under errors conditions, considering the applicability of the IEEE 802.15.4 standard in safety-critical environments such as industrial and vehicular.

## 1 INTRODUCTION

The applicability of wireless technologies on environments with temporal restrictions has been attracting interest of the real-time research community in the last decade (Åkerberg et al., 2011; Stone et al., 2012). The main advantages offered by wireless networks are: the reduction of size, weight, and power (SWaP) consumption; the ability to have mobile entities; and the possibility to establish networking communications where the use of wires is extremely difficult or impractical (Kandhalu and Rajkumar, 2012).

There are many studies in wireless communications addressing the temporal behaviour of communication services at the lowest level of the protocol stack (Han et al., 2011; Shuai and Zhang, 2010; Hou and Bergmann, 2010). These studies pay little or no attention to the dependability aspects of medium access control (MAC) sublayer and its services, which are essential to assure the timeliness and resilience of the network when operating under error conditions.

This paper evaluate the network simulator NS-2 to identify its limitations, proposing enhancements to provide a better knowledge and understanding of wireless network operation under such conditions. The NS-2 was chosen due to their native support to simulate wireless networks based on IEEE 802.15.4 standard, which is used as case study.

Our research achievements are organised as follows: Section 2 describes our system model, which comprises the assumptions utilised through the paper; Section 3 presents a brief overview of the IEEE 802.15.4 standard; Section 4 addresses the main temporal issues of the network operation under error conditions; Section 5 presents a brief overview of the NS-2 simulator, including its limitations; Section 6 presents the improvements in the IEEE 802.15.4 NS-2 module, including a fault injector that complements the existent NS-2 mechanisms, and a new component to perform the temporal analysis of the network operation; Section 7 presents the simulation setup of the Inaccessibility Scenarios and a simulation script description; Section 8 presents the results obtained in the simulation of IEEE 802.15.4 networks, allowing an enhanced temporal evaluation of such networks; Finally, Section 9 presents some conclusions and future directions of this work.

## 2 SYSTEM MODEL

Our system model is formed by a set of wireless nodes[1] $X = \{x_1, x_2, \ldots, x_n\}$, being $1 < n \leq \#A$, where $A$ is the set of all wireless nodes using the same communication channel. The set $X$ itself represents a network entity dubbed wireless network segment (WnS), as depicted in Figure 1. A WnS establishes a wire-

[1] A wireless node is a networked device capable to communicate with other nodes

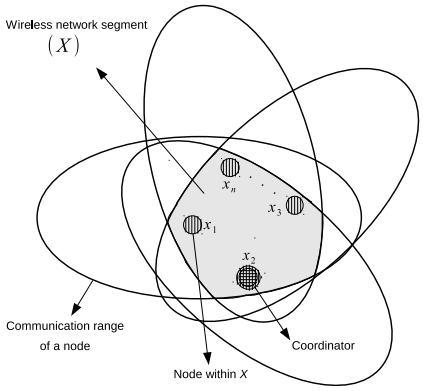


Figure 1: The graphical representation of a wireless network segment.

less network where each node can sense one another within one-hop of distance, being more complex networks composed by more than one WnS. For simplification purposes, our analyses assume a network with one WnS, being its behaviour supported by the following assumptions:

1. The communication range of $X$, i.e. its broadcast domain, is given by: $B_X = \bigcap_{j=1}^{n} B_D(x)$, $\forall x \in X$, where $B_D(x)$ represents the communication range of a node $x$;

2. $\forall x \in A$, $x \in X \iff B_D(x) \bigcap B_X = B_X$ or, as a consequence of node mobility, $x \notin X \iff B_D(x) \bigcap B_X \neq B_X$;

3. $\forall x \in X$ can sense the transmissions of one another;

4. $\exists x \in X$ which is the coordinator, being unique and with responsibility to manage the set;

5. A network component (e.g. a node $x \in X$) either behaves correctly or crashes upon exceeding a given number of consecutive omissions (the component's *omission degree*, $f_o$) in a time interval of reference[2], $\mathcal{T}_{rd}$;

6. failure bursts never affect more than $f_o$ transmissions in a time interval of reference, $\mathcal{T}_{rd}$;

7. omission failures may be inconsistent (i.e., not observed by all recipients).

For a given WnS, assumptions 1, 2, and 3 define the physical relationship between nodes, assumption 4 defines the existence of a coordinator, and assumptions 5, 6, and 7 define how the occurrence of communication errors are modelled and handled within the WnS. All communications and relations between nodes are established at MAC level, which are reinforced by assumption 3. As a consequence of mobility, nodes may be driven away of a given WnS (assumption 2). All communication errors within WnS are transformed into omissions (assumption 5), and in the context of network components an omission is an error that destroys a data or control frame.

[2]For instance, the duration of a given protocol execution. Note that this assumption is concerned with the total number of failures of possibly different nodes.

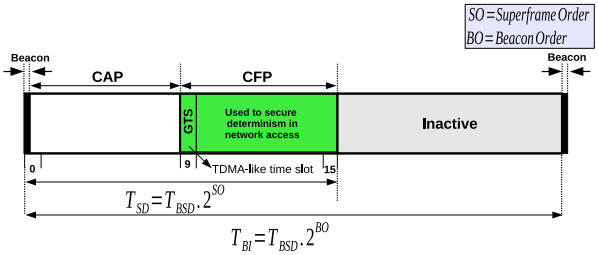


Figure 2: Superframe structure (Standard and Society, 2011)

# 3 IEEE 802.15.4 OVERVIEW

IEEE 802.15.4 is a standard for wireless sensor and actuator networks (WSANs), which support two operating modes: beacon-enabled and non beacon-enabled. In this paper we only address the beacon-enabled mode, which supports traffic with temporal restrictions. The network coordinator controls the network access through the superframe structure depicted in Figure 2.

This superframe structure comprises an active period, and optionally, an inactive period. In the active period there are a contention access period (CAP) and a contention free period (CFP). CAP is used to transmit traffic without any temporal guarantee and in a best effort approach. In CFP nodes can allocate time slots to transmit traffic with temporal restrictions (i.e., time division multiple access (TDMA) approach), where such slots are dubbed guarantee time slots (GTSs). The inactive period is used for power saving purposes (when needed).

As depicted in Figure 2, the duration of CAP and CFP are defined by two parameters: the beacon order (*BO*); and the superframe order (*SO*). The value of *BO* defines the superframe duration (i.e., the beacon interval, $\mathcal{T}_{BI}$), and the value of *SO* the duration

| Scenario | Equation |
|---|---|
| Single Beacon Frame Loss | $\mathcal{T}^{wc}_{ina\leftarrow sbfl} = \mathcal{T}_{BSD} \cdot (2^{BO} + 1)$ |
| Multiple Beacon Frame Loss | $\mathcal{T}^{wc}_{ina\leftarrow mbfl} = \mathcal{T}_{BSD} \cdot \left(2^{BO} + 1\right) \cdot nrLost$ |
| Synchronisation Loss | $\mathcal{T}_{ina\leftarrow nosync} = \mathcal{T}_{BSD} \cdot \left(2^{BO} + 1\right) \cdot nrLost$ |
| Orphan Node | $\mathcal{T}^{wc}_{ina\leftarrow orphan} = \mathcal{T}_{ina\leftarrow nosync} + \mathcal{T}_{MLA}(Orphan) + \sum_{j=1}^{nrchannels} \left(\mathcal{T}^{wc}_{MAC}(Orphan) + nrWait \cdot \mathcal{T}_{BSD}\right) + \mathcal{T}^{wc}_{MAC\_ack}(Realign)$ |
| Coordinating Realignment | $\mathcal{T}^{wc}_{ina\leftarrow realign} = \mathcal{T}_{MLA}(Realign) + \mathcal{T}^{wc}_{MAC\_ack}(Realign)$ |
| Coordinator Conflict Detection | $\mathcal{T}^{wc}_{ina\leftarrow C\_Detection} = \mathcal{T}^{wc}_{MAC\_ack}(C\_Conflict)$ |
| Coordinator Conflict Resolution | $\mathcal{T}^{wc}_{ina\leftarrow C\_Resolution} = \mathcal{T}_{MLA}(Conflict) + \sum_{j=1}^{nrchannels} \left[\mathcal{T}^{wc}_{MAC}(Beacon\_R) + nrWait.\mathcal{T}_{BSD}\right] + \mathcal{T}_{MLA}(Realign) + \mathcal{T}^{wc}_{MAC}(Realign)$ |
| Extract Request | $\mathcal{T}^{wc}_{ina\leftarrow extReq} = \mathcal{T}^{wc}_{MAC_{ack}}(ExtReq) + \mathcal{T}_{wait}$ |
| GTS request | $\mathcal{T}^{wc}_{ina\leftarrow GTS} = \mathcal{T}^{wc}_{MAC\_ack}(GTS)$ |
| Association | $\mathcal{T}^{wc}_{ina\leftarrow assoc} = \sum_{j=1}^{nrchannels} \left[\mathcal{T}^{wc}_{MAC}(Beacon\_R) + nrWait.\mathcal{T}_{BSD}\right] + \mathcal{T}_{MLA}(Beacon) + \mathcal{T}^{wc}_{ina\leftarrow extReq} + \mathcal{T}_{MLA}(AssocReq) + \mathcal{T}^{wc}_{MAC\_ack}(AssocReq)$ |
| Re-Association | $\mathcal{T}^{wc}_{ina\leftarrow reAssoc} = \mathcal{T}_{ina\leftarrow nosync} + \mathcal{T}^{wc}_{ina\leftarrow assoc}$ |

Table 1: Easy-to-use formulas defining the durations of periods of network inaccessibility

of the active period (CAP+CFP). The duration of the beacon interval is $\mathcal{T}_{BI} = \mathcal{T}_{BSD} \cdot 2^{BO}$, where $\mathcal{T}_{BSD}$ is the base value of $\mathcal{T}_{BI}$ when $BO = 0$, as defined within the IEEE 802.15.4 standard. The real length of CFP depends on the number of GTSs actually allocated. There is no inactive period when $BO = SO$, being the duration of the active period equal to $\mathcal{T}_{BI}$.

# 4 CHARACTERISING IEEE 802.15.4 NETWORK OPERATION UNDER ERROR CONDITIONS

The utilisation of IEEE 802.15.4 WSANs is emerging in environments such as industrial and vehicular, where some networking communications must respect restrict temporal constrains. Wireless communications may be affected by different sources of interferences, such as electromagnetic waves, obstacles on the communication path, or even by the mobility of nodes. Communication errors may occur as a consequence of such interferences, disturbing the communication services and the network operation itself.

The occurrence of such communication errors may affect two different types of operations, which are related to transmit data traffic and to control and maintain the network operation. The literature presents different works (Wang et al., 2012; Saifullah et al., 2011), which are only focused on the characterisation and presence of errors on data transmissions, disregarding the negative effects of such conditions in the MAC management operations.

In the context of MAC management operations, an already known severe consequence of communication errors is dubbed network inaccessibility (Souza and Rufino, 2009). A network inaccessibility period is characterised by the occurrence of "blackouts" within networking communications, where the network remains inaccessible by a temporary period of time. A research study performed by (Souza and Rufino, 2009) presents formulas to specify the duration of network inaccessibility for the IEEE 802.15.4 standard. Those communication "blackouts" may have a huge impact on the timeliness and dependability of the whole networking system, where a better evaluation may suggest the incompatibility of the guarantees offered by the communication service, and the temporal requirements of the target environment.

In Table 1 we present a summary of the worst case duration (represented by the superscript ($^{wc}$)) for each network inaccessibility scenario. As an example, we will briefly explain the characterisation of the most evident network inaccessibility scenarios, which are related with the loss of beacon frames. Three different inaccessibility scenarios may occur if such frames are not received correctly.

A single beacon frame loss occurs when only one beacon is lost. The duration of such scenario is equal to $\mathcal{T}_{BI} + \mathcal{T}_{BSD}$, where $\mathcal{T}_{BSD}$ is utilised as a temporal compensation to accommodate possible clock deviations between network nodes. The loss of multiple and consecutive beacons characterises the occurrence of the multiple beacon frame loss scenario, where a correct beacon is received after the loss of the previous *nrLost* beacons. The synchronisation loss is a special case of the multiple beacon frame loss scenario where after the loss of *nrLost* beacons the next beacon is also lost. The duration of both multiple beacon frame loss and synchronisation loss is a multiple of the single beacon frame loss, which is $nrLost \cdot (\mathcal{T}_{BI} + \mathcal{T}_{BSD})$. For simplification purposes we replace the $(\mathcal{T}_{BSD}.2^{BO})$ by $\mathcal{T}_{BI}$, as indicated in section 3. The complete network inaccessibility charac-

terisation for the IEEE 802.15.4 is present in (Souza and Rufino, 2009).

# 5 NS-2 SIMULATOR OVERVIEW

The NS-2 is a discrete-event simulation tool, widely used to study the dynamics of communication networks. It is developed in a collaborative effort by many institutions, containing contributions from different researchers.The simulation library and network protocols are written using the *C* and *C++* languages. The simulation environment is described and modified using the *OTcl* script language, without the necessity to recompile the whole NS-2 source code.

Every action in NS-2 is associated with events rather than time. An event comprises an execution time, a set of tasks, and a reference to the next event. These events are connected to each other, and form a chain of events on the simulation time line. The sequential execution of this chain of events is controlled and managed by a scheduler component, the brain and execution engine of the NS-2. It is possible to define its own procedures and variables to facilitate the interaction. The member procedures and variables in the *OTcl* domain are called instance procedures.

The IEEE 802.15.4 NS-2 module is provided in the form of methods of each layer specified in the IEEE 802.15.4 standard. The module came with different functionalities, and support different network topologies (star and point-to-point), two types of operation (beacon and non-beacon enabled), and basic MAC management actions such as Association, Channel Scan, energy model, etc.

# 6 ENHANCING NS-2 SIMULATOR

The evaluation of the network operation under error conditions needs components capable to inject faults in the simulation, which cause the network inaccessibility scenarios described previously. The NS-2 already provides components to perform such fault injection, but these components using an error model not capable to affect specific MAC frames, utilised by the IEEE 802.15.4 to control the access to the network.

To overcame the current error model limitation, we complement the existing NS-2 components with a new fault injector component, which is capable to generate faults in specific MAC frames. We also incorporate in NS-2 a a temporal analysis component,

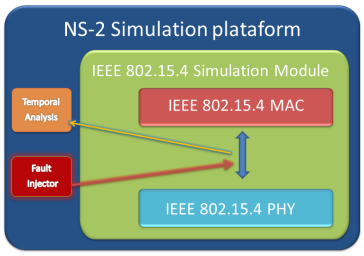


Figure 3: New Features in 802.15.4 module

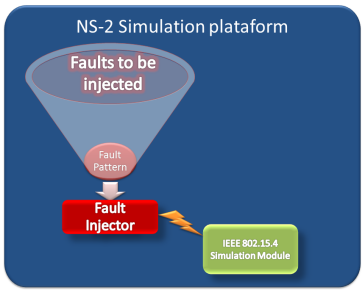


Figure 4: Fault Injector scheme

which is needed to account and measure the effects of faults generated by the fault injector component. These two components are independent of the type of network, being separated from the IEEE 802.15.4 module, as represented in the figure 3.

## 6.1 Fault Injector Component

Our fault injector is capable to use a fault pattern to inject errors during the simulation. The criteria to define the fault pattern is totally configurable, allowing the definition of deterministic or probabilistic fault patterns. An illustration of the fault injection scheme is shown in the figure 4.

A fault pattern can be defined to generate transmission errors randomly in time (random noise or interference) or be localized in specific time intervals (deterministic noise). On both of these patterns, the fault injector can be customized regarding the type of frame to affect, the rate and the duration of the fault injection.

Patterns with long duration are discouraged for deterministic error models, since such long duration may cause a permanent inaccessibility to the network access. For example, if we are corrupting a beacon frame injecting deterministic faults successively over a long period we may cause the loss of synchronization by the node and consequently this becoming unable to access the network again. However this type of pattern is beyond the scope of this work that is to analyse accidental faults where such pattern does not happen.

**Algorithm 1** Fault Injector - A random function

1: Begin.
2: $randomTime \leftarrow randomGenerator(seed)$;
3: $NewRandomEvent \leftarrow$ faultInjector(frameToCorrupt);
4: $Scheduler.schedule$(NewRandomEvent,randomTime);
5: $CorruptNode.Update()$;
6: End.

To perform the random noise or interference is possible to simulate aleatory errors on the network communication, injecting faults between the MAC and the PHY. A random function implemented in the fault injector allows inserting random corruption events in the NS-2 scheduler as described in Algorithm:1. In case of random noise the instant when the corruption occurs is totally aleatory, and is generated through a seed given by argument as described in line 2. A new event is created and the action associated with it is a frame corruption performed by the fault injector as indicated in line 3. Finally the *NewRandomEvent* which will perform the corruption is inserted in the NS-2 scheduler and executed at the defined instant as in line 4. An information about the corruption occurred in a specific node is recorded as described in line 5.

The fault injector achieves the frame corruption as described in Algorithm:2, accessing the command header of the frame as represented in line 5, and changing a bit in the frame content, implying the drop of these frames in the MAC level of the receiving nodes. When the frame is received if the fault injector is active, we can decide if a specific frame is affected or any frame that a node receives will be corrupted. The parameter $frameToCorrupt$ represented on line 3 is previously defined and if desired all the received frames can be affected defining the $frameToCorrupt$ to a specific value. An information about the corruption occurred in a specific node is recorded as described in line 6. This information is used for a better control of the simulation events. The fault injection may be performed in the coordinator, which implies, depending on the type of frame affected, that the whole network may be inaccessible, in the specific case of affecting a MAC control frame. In case we decide to affect a MAC control frame, affecting specific network points, the fault injection can be performed for example at non-coordinator nodes tracking the reception of beacon frames. In the specific case, when we perform corruption in a MAC control frame such as the beacon in the coordinator, none of the nodes receives the beacon and therefore the whole network will be inaccessible. Otherwise, when the corruption is performed in the nodes that should receive beacon frames, only the node that has

**Algorithm 2** Fault Injector Mechanism

1: Begin.
2: $MAC.Receive(frame)$;
3: **if** $frame = frameToCorrupt$ **then**
4: **for** selected Fault Pattern **do**
5: $CommandHeader(frame) -> error() = 1$;
6: $CorruptNode.Update()$;
7: **end for**
8: **end if**
9: End.

the fault injector component activated, i.e. beacon corruptions occurring, cannot access the medium and becomes inaccessible. The corruption of the frames can be disabled, through the deactivation of the fault injector on the *tcl* script, and the normal behaviour of the network restored at any time.

## 6.2 Temporal Analysis Component

The temporal analysis component was designed to measure the temporal aspects of received frames, from their duration to the effects of frames received with errors during the network operation. In this paper, we instrumented the temporal analysis component to measure network inaccessibility events. As the occurrence of network inaccessibility is related with the MAC control frames (e.g., beacons), this component was configured to monitor and capture information about such kind of frames.

Let us present an example to characterise how the temporal analysis component works. We use the beacon frame as the frame to be monitored in such example. When the received beacon is corrupted, the temporal analysis component starts a timer to account the related network inaccessibility period. When a new beacon is received successfully, the timer is stopped and the duration of such period is registered.

When the simulation is finished, the temporal analysis component generates a report, regarding all events captured during the simulation. The report is utilised as input to a *gnuplot* script, which transform the raw data within the report to a graphical representation of the captured events.

# 7 SIMULATING INACCESSIBILITY SCENARIOS

The simulation is defined in an OTcl script (Listing:1) and is carried out in an one-hop star topology, where all the nodes are within the range of each

other.

```
1 Event at 0.0   node_(0)
    startWnSCoordinator $beaconOrder
    $superFrameOrder";
2 Event at 20.0 node_(1) & node_(2)
    startDevice"
3 Event at 20.0 node_(1)
    enableTemporalAccount $Scenario";
4 Event at 30.0 node_(0)
    startBeaconTransmission
    $beaconOrder $superFrameOrder"
5 Event at 30.0 node_(1) GTS On"
6 Event at 30.0 node_(1) Start Fault
    Injection $Beacon $Rounds"
7 Event at $stopTime "stop"
```

Listing 1: Example of NS-2 Simulation Script

In the script (Listing:1) we define that the first node to start was the coordinator, specifying the values of its *BO* and *SO* in line 1. After the WnS is established we start the nodes in line 2. Our temporal analysis component is enabled on line 3, given the selected scenario. The periodic beacon transmission is initiated at the coordinator on line 4, taking the *BO* and *SO* as arguments. At line 5 we enable the GTS transmission for the node(1), which means that each time this node have data to transmit will use the GTS mechanism. Finally, at line 6 we start our fault injector to, in this example, corrupt beacon frames for a certain number of rounds.

To simulate a network operation under error conditions, implying the occurrence of network inaccessibility, we configure our fault injector component to generate deterministic faults. For each addressed scenario we set our fault injector to corrupt a specific frame at a given number of times, on a chosen node. The fault injector can corrupt one of each frame type present in the Table: 2.

To achieve the **Single Beacon Frame Loss (SBFL)** scenario we executed the following schedule of Events:

```
1 Event at 30.0 node_(1) Start Fault
    Injection $Beacon $SBFL"
```

In this scenario the beacon frame will be corrupted *SBFL* number of times (i.e., only one time) at the Node(1), after 30 simulation seconds.

The **Multiple Beacon Frame Loss (MBFL)** occurs when we change the number of corrupting rounds on the fault injector depending on the value that *MBFL* assumes in order to achieve the loss of *nrLost* beacons. The **synchronization loss** is a special case of the MBFL scenario where after the loss of *nrLost* beacons the next beacon is also lost.

| Frame type value | Command frame ID | Standard Reference |
|---|---|---|
| 0 | | Beacon |
| 1 | | Data |
| 2 | | Ack |
| 3 | | MAC Control Frame |
| | 01 | Association Request |
| | 02 | Association Response |
| | 03 | Disassociation |
| | 04 | Data Request |
| | 05 | Coordinator conflict |
| | 06 | Orphan |
| | 07 | Beacon request |
| | 08 | Coordinator realignment |
| | 09 | GTS request |

Table 2: MAC frame types

```
1 Event at 30.0 node_(1) Start Fault
    Injection $Beacon $MBFL"
```

The **Orphan notification and Coordinator realignment** are achieved when the fault injector corrupts *NOSYNC* beacon frames, corresponding to the current scenario, and the node lose the synchronization. The Orphan notification is observed on the node and the Coordinator realignment is transmitted by the coordinator on response.

```
1 Event at 30.0 node_(1) Start Fault
    Injection $Beacon $NOSYNC"
```

So that **Coordinator Conflict Detection** can occur, this event has to be forced on the simulator. Once every time a node becomes a coordinator it assumes its ID as the *networkID*, so a coordinator conflict is impossible because every coordinator assumes a distinct ID. To force that event we oblige the coordinator to use the same identifier with the following line.

```
1 Event at 0.0   node_(1) Coordinator
    Conflict 1"
2 Event at 0.0   node_(0) Coordinator
    Conflict 1"
```

When the **GTS mechanism** is previously activated from the script, and the node has data to transmit, a GTS Request will occur. This request will be send to the coordinator by the node to perform an allocation of a GTS slot for exclusive transmission time.

```
1 Event at 30.0 node_(1) GTS On"
```

| Simulation Parameters | |
|---|---|
| NS-2 Version | 2.35 updated with GTS, Fault Injector, and Temporal Analysis features |
| Network Topology | Star Topology |
| Nodes | 7 |
| Traffic | Constant Bit Rate (CBR) |
| Reception range | 15m |
| Carrier Sense range | 15m |
| Packet Size | 8, 67, 127 kbytes |
| CAP Transmission Type | Direct, using CSMA/CA |
| CFP Transmission Type | GTS transmission |
| Transmission/Reception Power | 30mW |
| Beacon | Enabled |
| Beacon Order | 3 |
| Superframe Order | 3 |
| Maximum CSMA/CA Attempts | 4 |
| Simulation Time | 600 seconds |

Table 3: Simulation Parameters

# 8 RESULTS

## 8.1 Simulation Setup

The network was simulated with seven nodes, where one of these nodes, in the center, was the coordinator. All other nodes are in the radio transmission range of the coordinator, and in the range of each other. A $BO = 3$ was utilised to specify the superframe duration within simulations.

The characteristics of the simulation setup scenario are shown in Table 3. To evaluate the network behaviour under error conditions we applied different error patterns on the simulation through fault injection.

The fault injection can be performed using three different durations: short, medium, or long. The short had the duration of a normal frame transmission, the medium had the duration of the transmission of 3 frames, and the long had the duration of half a beacon.

The traffic generator is set to produce Constant Bit Rate traffic (CBR), which means data frames are transmitted at a constant rate from the nodes to the coordinator. The payload of the sent data was also varied, being the smallest payload of 8 kilobytes, the medium of 67 kilobytes, and the large of 127 kilobytes. The characteristics of the simulation scenario are shown in Table 3.

## 8.2 Simulation Results

After the environment setup, the simulation was performed to obtain the best and worst case duration of the inaccessibility scenarios.

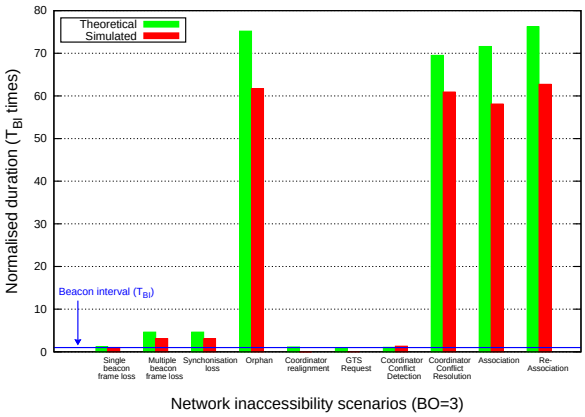


Figure 5: Normalized Inaccessibility Scenarios comparison between Theoretical and Simulated worst case (BO=SO=3 and $\mathcal{T}_{BI} = 0.123$s)

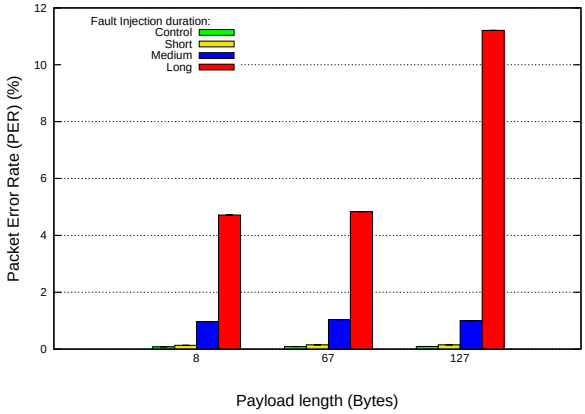


Figure 6: Packet Error Rate comparison between different error patterns

Figure 5 shows the graphic that represents the duration of each network inaccessibility scenario, comparing the results of the previous theoretical study in (Souza and Rufino, 2009) and the obtained simulation results. The results presented in Figure 5 clearly shows that periods of inaccessibility may have a huge duration, which represents a non-negligible impact for networks with temporal restricted traffic.

In the Figure 6 we can observe the Packet Error Rate (PER) between different error conditions. As expected, longer periods of fault injections implies a higher PER. In comparison with the Control frame, a frame that was transmitted without errors, we can see an increasing PER related to the higher periods of fault injection.

The result with the greatest impact is related with the bigger payload, achieves a PER of more than 10%.

An important result of this study is that the influence of network errors, causing periods of inaccessibility in the network, cannot be overlooked if pre-

dictability and real-time operation is a system requirement, under the risk of jeopardize the safety and timeliness of the entire system. The effects of network inaccessibility incidents should be controlled by the definition of strategies for the reduction of the periods of inaccessibility, which is achieved in (Souza and Rufino, 2013).

# 9 CONCLUSION AND FUTURE DIRECTIONS

The paper addressed the behaviour of IEEE 802.15.4 networks in the presence of network errors, leading to periods of network inaccessibility.

Significant improvements and modifications in the NS-2 simulator IEEE 802.15.4 module were presented, which includes the specification of two additional components capable to inject and measure the effects of errors during the network operation. The presence and use of these two component were essential to perform the simulation and evaluation of all network inaccessibility scenarios.

The results obtained by our simulations evidence the relevant temporal aspects of the IEEE 802.15.4 beacon-enabled networks operating under error conditions. Such results can be utilised in the specification of a robust timeliness model of the network, in order to achieve an effective support to real-time operation in IEEE 802.15.4 networks.

## Acknowledgements

This work was partially supported: by the EC, through project IST-FP7-STREP-288195 (KARYON); by FCT/DAAD, through the transnational cooperation project PROPHECY; and by FCT, through the project PTDC/EEI-SCR/3200/2012 (READAPT), the Multiannual Funding Program, and the Individual Doctoral Grant SFRH/BD/45270/2008.

# REFERENCES

Han, S., Zhu, X., Mok, A., Chen, D., and Nixon, M. (2011). Reliable and Real-Time Communication in Industrial Wireless Mesh Networks. *2011 17th IEEE Real-Time and Embedded Technology and Applications Symposium*.

Hou, L. and Bergmann, N. (2010). System Requirements for Industrial Wireless Sensor Networks The University of Queensland.

Kandhalu, A. and Rajkumar, R. (2012). Qos-based resource allocation for next-generation spacecraft networks. In *IEEE 33rd Real-Time Systems Symposium (RTSS)*.

Åkerberg, J., Gidlund, M., and Björkman, M. (2011). Future research challenges in wireless sensor and actuator networks targeting industrial automation. In *9th IEEE Internation Conference on Industrial Informatics (INDIN)*.

Saifullah, A., Xu, Y., Lu, C., and Chen, Y. (2011). End-to-end delay analysis for fixed priority scheduling in wirelesshart networks. In *17th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*.

Shuai, X. and Zhang, Z. (2010). Research of Real-Time Wireless Networks Control System MAC Protocol. *Journal of Networks*.

Souza, J. L. R. and Rufino, J. (2009). Characterization of inaccessibility in wireless networks - a case study on IEEE 802.15.4 standard. In *IFIP 3th International Embedded System Simposium IESS*.

Souza, J. L. R. and Rufino, J. (2013). Analysing and reducing network inaccessibility in IEEE 802.15.4 wireless communications. In *38th IEEE Conference on Local Computer Networks (LCN)*.

Standard, I. and Society, I. C. (2011). *IEEE Standard for Local and metropolitan area networks, Part 15 . 4 : Low-Rate Wireless Personal Area Networks ( LR-WPANs ) IEEE Computer Society S ponsored by the*.

Stone, T., Alena, R., Baldwin, J., and Wilson, P. (2012). A viable COTS based wireless architecture for spacecraft avionics. In *IEEE Aerospace Conference*, pages 1–11.

Wang, J., Dong, W., Cao, Z., and Liu, Y. (2012). On the delay performance analysis in a large-scale wireless sensor network. In *IEEE 33rdReal-Time Systems Symposium (RTSS)*.