WiSH-WalT: A Framework for Controllable and Reproducible LoRa Testbeds

Andrzej Duda, Qasim Lone, Etienne Duble, Franck Rousseau, Ingrid Moerman, Spilios Giannoulis

WiSH-WalT: A Framework for Controllable and Reproducible LoRa Testbeds

Qasim Lone*, Etienne Dublé*, Franck Rousseau*, Ingrid Moerman¶,
Spilios Giannoulis¶, and Andrzej Duda*
*Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, F-38000 Grenoble, France
Email: firstname.lastname@imag.fr
¶Department of Information Technology, Ghent University—imec, IDLab, B-9052 Ghent, Belgium
Email: firstname.lastname@ugent.be

***Abstract*—In this paper, we describe the design and implementation of WiSH-WalT, a framework for controllable and reproducible LoRa testbeds. The implementation of the WiSHFUL unified interface provides the means for controlling and adapting the LoRa parameters to given conditions and low energy consumption by configuring transmission power, spreading factor, bandwidth, and error coding rate. WalT provides support for running reproducible experiments based on the deployment of docker images on a distributed set of nodes. Put together, the functionalities of WiSHFUL and WalT open new possibilities for deploying reproducible LoRa testbeds. We have used WiSH-WalT to experiment with adaptive configuration of the LoRa parameters and evaluated the communication performance of LoRa motes in a setup with several gateways.**

I. INTRODUCTION

This paper describes the design and implementation of a framework for controllable and reproducible LoRa testbeds. Reproducibility of experiments is an important part of the research methodology in networking. Specialized testbeds such as ORBIT [1] or IoT-LAB [2] make the experimentation task easier and offer good support for experiment repeatability. However, they only offer operating conditions that are far away from real-world deployments: motes are usually distributed on a regular grid in one large room, so reproducibility in varying environmental conditions is limited. The long range of LoRa networks pushes wireless lab testbeds to the limits—you cannot set up a meaningful experiment in a lab.

For *reproducible network experiments*, we developed WalT [3]. WalT nodes are single-board computers (Raspberry Pi, RPI) on which users can deploy their OS (filesystem, kernel) packaged as a *docker* image for easy customization and sharing. WalT nodes powered by PoE Ethernet switches interconnect the wireless nodes under test and can gather traces of operation to understand the complex behavior of all interacting entities and measure their performance. With low-cost small-sized standard components and free software, researchers can easily reproduce their own WalT platform to validate results in real-world conditions. With WalT, a researcher can: i) easily setup a platform to develop, debug, and validate protocols and applications and (ii) deploy and control experiments on a larger scale at a target location. Reproducibility of a WalT testbed means that other researchers can deploy and repeat exactly the same experiment in a different environment.

The WiSHFUL project [4] also addresses the issue of network experiments with complementary objectives: i) abstract hardware platforms with a unified interface to easily control radio and network settings, ii) provide a global control framework for dynamic configuration, and an intelligence framework to enable autonomous control strategies based on machine learning, and iii) support real-life environments with compact portable testbeds.

WiSHFUL offers several software platforms for various wireless networks with data plane and control plane functionalities for advanced and intelligent radio and network control. The proposed unified radio control (UPI or Unified Programming Interface) provides an abstraction of hardware specific instructions to enable flexible radio configuration, while the unified network control allows rapid prototyping and adaptation of network protocol stacks in a heterogeneous, multi-vendor environment.

In this paper, we report on WiSH-WalT[1] offering an integrated framework for LoRa experiments based on the implementation of the WiSHFUL unified interface on LoRa STMicroelectronics boards and its integration within WalT, which results in a framework offering the best functionalities of both systems. The UPI interface provides a flexible way for configuring the main parameters of the LoRa transceiver and for getting statistics gathered on the board. It allows controlling and adapting the LoRa parameters to given conditions and required energy consumption. WalT brings easy deployment of LoRa motes and monitoring of their operation. Put together, the functionalities of WiSHFUL and WalT open new possibilities for deploying reproducible LoRa testbeds.

The rest of the paper discusses related work (Section II), describes the properties of LoRa networks (Section

[1]available at https://gricad-gitlab.univ-grenoble-alpes.fr/Drakkar-LIG/WalT/wikis/home.

III), presents the implementation of the WiSH-WalT functionalities (Section IV), and reports on experiments with adaptive configuration of LoRa parameters in a network with several gateways (Section V).

II. Related Work

To study the influence of LoRa parameters on performance, we can either rely on analytical modeling [5], [6], simulation [7], [8], [9], [10], [11], [12], or measurements [9], [13], [11], [14].

Augustin et al. used simulation to evaluate the collision ratio for an increasing number of contending devices [9]. The results confirm the behavior of ALOHA with the maximum channel capacity of 18% for a link load of 0.48. At this load, there are around 60% of packets dropped. The problem with this evaluation is the fact that it does not take into account the *capture effect*.

Haxhibeqiri et al. [11] used a simulation model based on the measurements of the interference behavior between two motes with a duty cycle of 1% to show that when their number increases to 1000 per gateway, the packet loss rate increases to 32%. However, this level of the loss rate should be considered as low compared to 90% in pure ALOHA for the same load and it results from taking into account the capture effect, which apparently plays an important role in the LoRa performance.

The same study by Augustin et al. [9] presented throughput measurements on a LoRa testbed in addition to the simulation of the ALOHA behavior showing: i) less than 10% of loss rate over a distance of 2 km for SF (Spreading Factor) 9-12 and ii) more than 60% of loss rate over 3.4 km for SF 12.

Petrić et al. [13] measured the performance of LoRa in an urban setup. They showed a highly variable packet error rate (between 3 and 90%) for the range of 3 km from a gateway under the following conditions: bandwidth of 125 kHz, coding rate of 4/5, transmission power of 14 dBm, and spreading factor SF 7, 9, and 10. They also note that the correlation between RSSI (Received Signal Strength Indicator), SNR (Signal to Noise Ratio), and packet error ratio is not straightforward.

Bor and Roedig presented an analysis of the impact of LoRa transmission parameter selection on communication performance and energy consumption based on measurements of a 50 m LoRa link in an indoor setting [14]. They also developed a link probing scheme to determine transmission settings that satisfy performance requirements.

With respect to the testbed platforms such as WiSH-WalT, OpenChirp is an architecture that comes close to our work [15]. The framework provides tools to reconfigure the LoRa gateway, it supports user registration of LoRa devices, and obtaining parameters through gateways. Compared to OpenChirp, we propose more advanced functions that allow for better deployment, control, and supervision of LoRa experiments.

III. Overview of LoRa

LoRa has become an interesting technology for lightweight smart sensing in the Internet of Things (IoT) [16]. It defines a specific radio layer based on the Chirp Spread Spectrum (CSS) modulation and a simple MAC channel access method called LoRaWAN [17]. We can control the physical layer of LoRa through the following parameters [18]:

- Bandwidth (BW) – it is the range of transmission frequencies. We can configure the bandwidth between 7.8 kHz and 500 kHz. A larger bandwidth allows for a higher data rate, but results in lower sensitivity.
- Spreading Factor (SF) - it represents the ratio between the symbol rate and the spreading spectrum chip rate: there are 2^{SF} chips per symbol for a given value of SF. SF varies between 6 and 12 with SF 12 resulting in the highest sensitivity and range, but achieving the lowest data rate and increased energy consumption. The decrease of 1 in SF doubles the transmission rate and divides by 2 the transmission duration as well as energy consumption.
- Coding Rate (CR) – it corresponds to the rate of Forward Error Correction (FEC) applied to improve packet error rate in presence of noise and interference. A higher coding rate results in better robustness, but increases the transmission time. The possible values are: 4/5, 4/6, 4/7, and 4/8.
- Transmitted Power (TP) – LoRaWAN [17], the MAC layer for LoRa, defines the following values of TP for the EU 863-870 MHz band: 2 dBm, 5 dBm, 8 dBm, 11 dBm, and 14 dBm.

A given combination of the parameters results in a specific data rate and sensitivity that determines the transmission range.

The achievable data rates depend on the chosen bandwidth, spreading factor, and coding rate: a higher bit rate results from lower SF, higher BW, and CR of 4/5, at the cost of lower sensitivity and range. The bit rates range from 250 b/s to 11 kb/s: 250 b/s corresponds to SF 12 for BW of 125 kHz, whereas 11 kb/s results from SF 7 over BW of 250 kHz. Motes can also use the FSK modulation to reach a higher data rate of 50 kb/s. Table I presents SF, the data rate, SNR limit, and the airtime for a 10 byte packet.

Sensitivity ranges from -136 dBm for SF 12 (long range) and BW of 125 kHz to -111 dBm for SF 6 and BW of 500 kHz (shorter range). Depending on the transmitted power and sensitivity, the theoretical transmission range (assuming the Okumura Hata propagation model for an open rural environment) may vary from 6.5 km

SF	Chirps/symbol	SNR limit	Airtime	Bit rate
7	128	-7.5 dB	56 ms	5469 b/s
8	256	-10 dB	103 ms	3125 b/s
9	512	-12.5 dB	205 ms	1758 b/s
10	1024	-15 dB	371 ms	977 b/s
11	2048	-17.5 dB	741 ms	537 b/s
12	4096	-20 dB	1483 ms	293 b/s

Table I: LoRa parameters for BW of 125kHz.

for SF 12 and TP of 14 dBm to 1.3 km for SF6 and TP of 2 dBm [10].

We proposed optimal strategies for adapting the parameters from the point of view of energy efficiency [19]. For instance, we showed that the best SF and TP configuration to achieve minimal energy consumption (for BW of 125 kHz and CR of 4/5) consists of 1) increase TP from 2 dBm until reaching TP of 14 dBm with SF 6 and then 2) adapt SF until reaching SF 12 with TP of 14 dBm.

For class A devices (those that may achieve long lifetimes through the use of low duty cycles), LoRaWAN [17] defines an access method to the radio channel similar to ALOHA: a device wakes up and sends a packet to a Gateway right away. The difference with pure ALOHA is the variable packet length in LoRa. This choice of the access method highly impacts the capacity of LoRa and its scalability to a large number of devices. The scalability of LoRaWAN is first related to the ERC limitation of the duty cycle to 1% in the 868 MHz band for motes operating as ALOHA. For instance, with SF 12 and 100 motes, each of them may send 546 packets per day, or if there are 1000 motes, they can send 54 packets per day. The gateway can only send 9 ACK packets per mote to 100 motes under this assumption. These constraints have led to the Fair Access Policy in The Things Network (TTN) [20]: a mote has the limit of 30 s of airtime (19 packets per day for SF 12) and 10 downlink packets per day (ACKs).

We proposed a simple enhancement to LoRaWAN that does not impact energy consumption—the CSMA principle consisting of testing the channel if it is used by another transmission before attempting to send a packet [12].

LoRaWAN defines the Adaptive Data Rate (ADR) algorithm run on the gateway to control the data rate of motes. The gateway estimates the SNR level of the last 20 packets of a mote. It then chooses the data rate and TP suitable for the given level of SNR, and sends the parameters in the `LinkADRReq` frame to the mote. Motes can either enable or disable ADR based on their constraints.

IV. WiSH-WalT Implementation

WiSH-WalT uses the WalT platform to manage the deployment of LoRa motes, while WiSHFUL takes care of the local control of LoRa motes for configuring their communication parameters. Figure 1 provides a general view of the WiSH-WalT architecture. The WalT platform includes the WalT server and WalT nodes that can directly run on Raspberry Pi (RPI) (physical WalT nodes) or on a virtual machine (virtual WalT nodes running on the WalT server). RPI under WalT connects to LoRa motes or to a LoRa gateway via a serial interface.

A. WalT for Deployment and Initial Configuration

WalT provides:

- Lightweight and portable tool for exploring the topology, interacting with nodes, deploying a given operating system image on them, customizing such an image, and exploring logs. The user executes commands on the WalT server, the controller for *WalT nodes*.
- Easy OS customization for WalT nodes. Each operating system is packaged as a docker image (called *WalT image*), which allows modification of images in a docker container. For instance, we have modified some existing images to include the WiSHFUL controllers (Global and Local Control Programs).
- Raspberry Pi (RPI) acting as WalT nodes. Once they are installed with the WalT network bootloader, they are detected by the WalT server and registered as WalT nodes.
- Seamless and instant deployment of WalT images on WalT nodes that control LoRa nodes. What we call *deployment* is actually a customized network boot procedure: the server exposes the content of a WalT image (kernel, device-tree, and filesystem) as an NFS share, and it lets the selected nodes boot this image over the network.

The configured WalT images can be shared by publishing them on the *docker hub*. For instance, it allows us to share the images pre-configured with the WiSHFUL controller. Finally, WalT users can control the whole process remotely. A command line utility provides all options needed to control WalT nodes (deployment of a WalT image, remote shell, file transfer, etc.). Since we have added scripts to the deployed WalT image with the WiSHFUL controller, we can also fully control a LoRa mote connected to the USB port of a WalT node (flash, reboot, shutdown, and log of its output).

B. WiSHFUL for Controlling LoRa Motes

WalT images include the WiSHFUL Local Control Program so we can deploy them on RPIs connected to motes. We can install the WiSHFUL Global Control Program either on a physical machine (RPI) or run it on a virtual WalT node. The Local Control Program implements the WiSHFUL UPI for LoRa mote configuration (DevAddr, AppKey) and setting/getting the transmission parameters (transmission power, coding rate, and data rate) of LoRa motes. Based on the functions related to

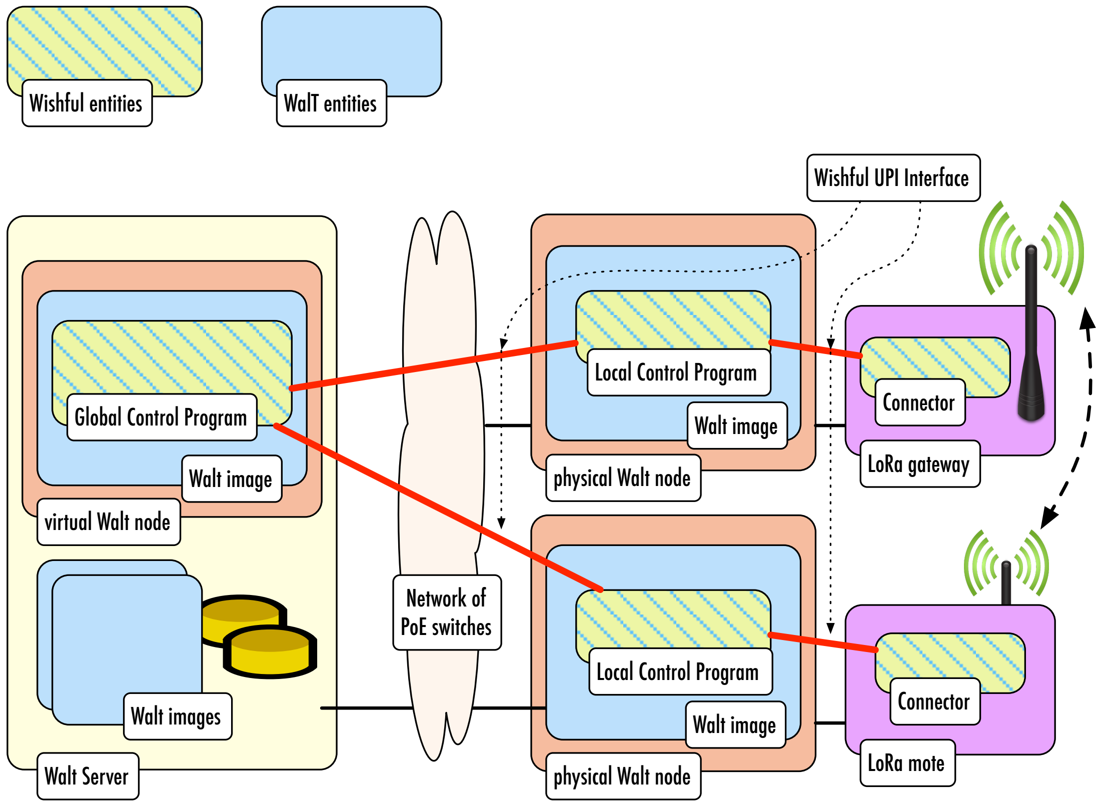

Figure 1: Architecture of WiSH-WalT

the identifiers and keys, WiSH-WalT provides a script to configure a LoRa device to work in the TTN network with one command.

We have developed the connector to the WiSHFUL unified interface on LoRa motes (STM32L03RZ Nucleo Cortex M0+ development board with Semtech SX1276) and integrated it with the WiSHFUL Local and Global Control Programs for managing LoRa experiments. The connector parses parameters sent by the Local Control Program and executes the required functions to change parameters or obtain their values. For communication between RPI and the LoRa motes, we have modified the iCube application to send and receive data on the serial port.

As mentioned previously, the ERC recommendations impose the duty cycle of 1% in the European 863-870 MHz ISM band. To save battery lifetime, the iCube LoRaWan application goes to sleep for a backoff period until a new time slot is available to send data. The WiSHFUL Local Control Program continues to send data to the LoRa mote until it receives an acknowledgment back for the successful change (or reception) of a parameter.

C. WalT Adaptation for Remote Nodes

As LoRa motes can span long distances, we have looked for the ability to control WalT deployments over wide areas through a VPN. However, WalT nodes heavily rely on LAN operation because of their network boot

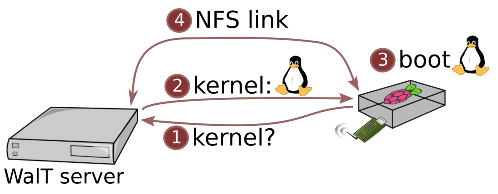

Figure 2: Boot procedure of a local physical WalT node

procedure (see Figure 2). Actually, booting a node over a VPN would mean that its network bootloader should itself connect to the VPN, then download and run the Linux kernel over this VPN connection, which is well beyond the abilities of current network bootloaders. We have introduced a level of virtualization in the node boot procedure (see Figure 3). Instead of just a network bootloader, remote WalT nodes have a complete operating system. Considering RPI nodes, this operating system is installed on their SD card (whereas classical WalT nodes just have the network bootloader). The operating system first sets up the local network connection (depending on the network where the node is installed), then connects to the VPN. At this time, a `kvm` virtual machine is started with its virtual network interface connected to the VPN. Any USB device connected to the physical node (such as a LoRa node) is also attached to this virtual machine. Then, the virtual machine acts exactly as a classical WalT node: it uses the classical WalT network bootloader to interact with the server and boot the selected WalT image.

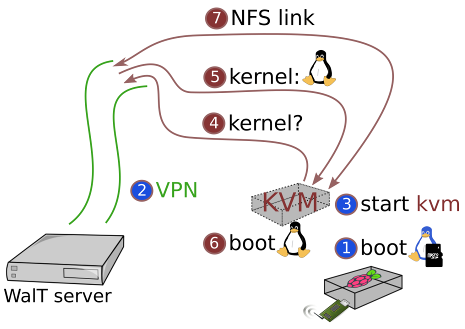

Figure 3: Boot procedure of a remote virtual WalT node

V. Experiments

We have set up a series of experiments with a LoRa mote sending packets to several public TTN gateways (see Table II for the information on the place of nine involved gateways).

# Gateway	Altitude (m)	Distance (km)
1	236	0.136
2	249	1.062
3	241	1.127
4	248	1.867
5	253	2.029
6	246	2.634
7	246	3.260
8	248	4.019
9	2253	13.192

Table II: Public TTN gateways used in the experiments

We wanted to test WiSH-WalT in real-world conditions and explore the problem of choosing the right gateway for a mote operating in a given place in function of different parameters. To get packet statistics from the TTN gateways, we have prototyped a script running on a virtual WalT node that queries the TTN API for getting the results of experiments and uses the WalT logging support to store them.

In the experiment, the LoRa mote was sending 50 packets per parameter (SF and TP). LoRa gateways are usually multi-channel so they can simultaneously demodulate on multiple channels. We have used the frame counter value to differentiate between the frames and have retained the maximal SNR value for the same frame demodulated using different channels. We have computed PDR from the number of packets received per gateway for every series of 50 packets sent. A round of measurements cycled over different parameters so that the transmission of two consecutive packets had different configuration parameters. As we only use WiSH-WalT to configure the transmission parameters on the LoRa mote and not for sending packets, WiSH-WalT does not have any impact on the transmission performance.

We present the statistics on the main performance indicators: SNR and PDR for CR of 4/5 and BW of 125

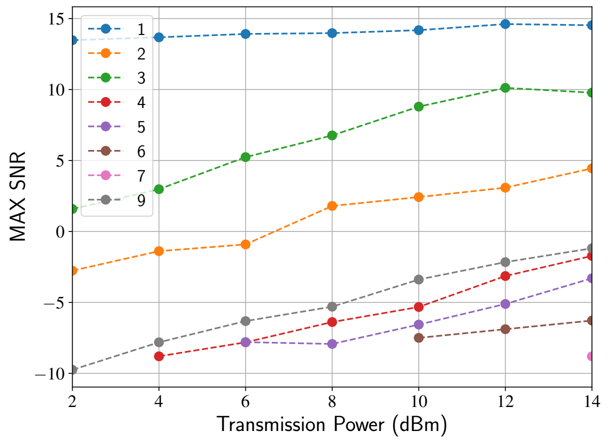

Figure 4: SNR in function of TP (SF 7)

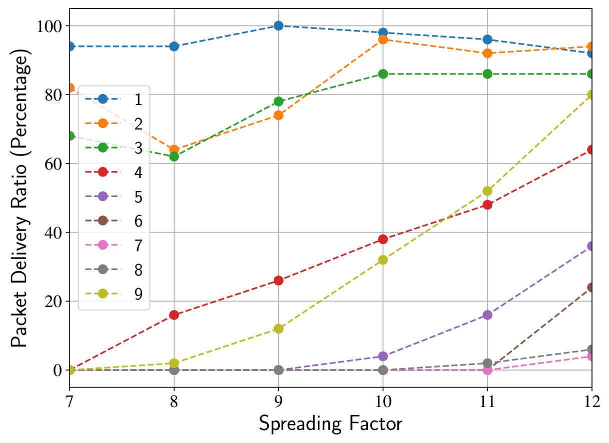

Figure 5: PDR in function of SF (TP 2 dBm)

kHz. We have also gathered RSSI, but it does not provide more insight into the observed LoRa performance.

For SNR in function of SF, we have observed that SF does not impact much SNR (and yet increasing SF by one adds 2.5 dB to the link budget [18]) and the level of SNR follows the ranking of distances except for Gateway 9 (figure not included for space reasons).

Figure 4 presents SNR in function of TP. For low values of SNR, decreasing SF results in lost connectivity with some gateways (e.g., Gateways 5, 6, 7, 8) well in line with Table I. Increasing TP results in better SNR except for the closest Gateway 1 that already benefits from a stable high value of SNR. Gateway 9 benefits from higher SNR than some other closer gateways because of its high point position and a clear line of sight.

Figures 5–8 show PDR in function of SF and TP while Figure 9 presents temporal variation of PDR during a day. We can draw up the following remarks:

- Gateway 1 benefits from a high quality of reception with almost no impact of SF and TP.
- We observe an anomaly for Gateways 1, 2, and

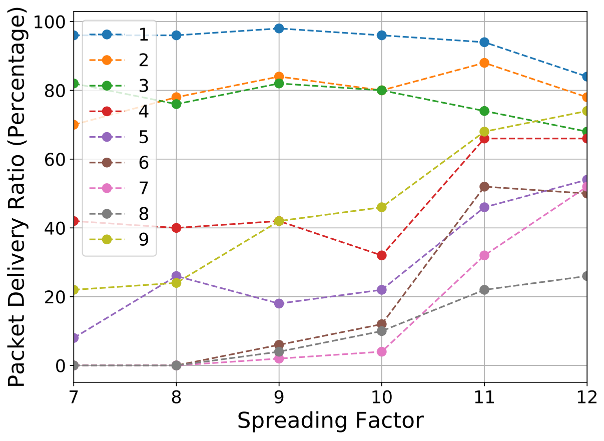

Figure 6: PDR in function of SF (TP 14 dBm)

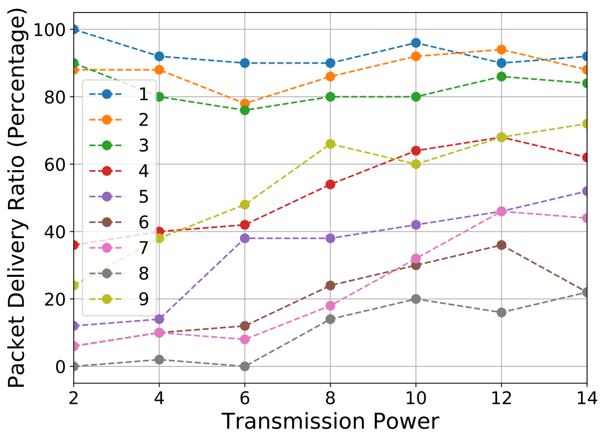

Figure 7: PDR in function of TP (SF 12)

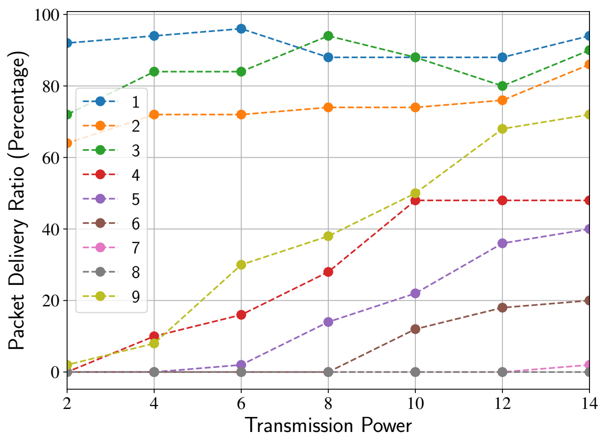

Figure 8: PDR in function of TP (SF 7)

3 with PDR going down for increasing SF, which probably comes from contention: the collision probability is higher for longer transmission times.

- For other Gateways, the impact of SF and TP is as we could expect—we obtain better PDR with larger SF and TP. However, a good level of PDR (greater or equal to 80%) is only achieved for Gateway 1 as well as Gateways 2 and 3 for some parameter values

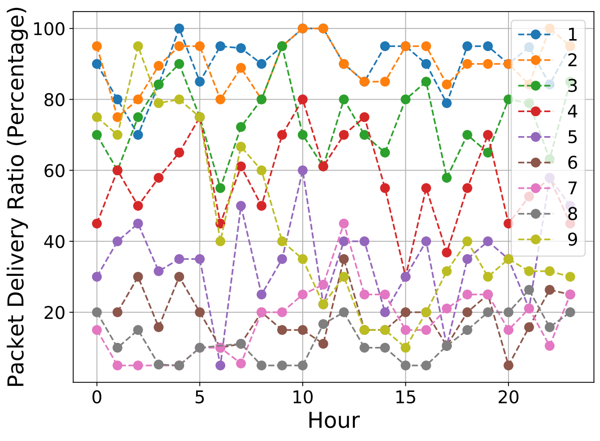

Figure 9: PDR during a day (SF 12, TP 2 dBm)

(SF = 10-12, TP = 2 dBm). They are relatively close to the mote (< 1.2 km).

- A very good level of SNR does translate into a very good level of PDR. However, for some gateways, good levels of SNR result in average PDR, e.g., Gateway 4 only obtains between 0 and 50% PDR, which may come from obstacles in the line of sight or from increased contention. Other studies reported similar conclusions [13]. The measurements also show that ADR, the current bit rate adaptation algorithm based on SNR is probably not the best way to cope with changing conditions.
- A high position of a gateway is important because it brings a clear line of sight: Gateway 9 in the range of 13 km obtains good PDR even for this relatively long distance.
- Nevertheless, variations in PDR are important: Figure 9 shows the average differences of around 40 % and even more for Gateway 9—it changes between 95 % and 10 % (data gathered on April 15, 2018 with some cloudy and windy weather in the afternoon).

Energy	t_{TX} [ms]	P_{TX} [mA]	$E = t_{TX} P_{TX}$ [mJ]
SF8-TP8	103	59	6.077
SF7-TP14	56	92	5.152

Table III: Energy consumption for different parameters

Finally, we wanted to experiment with an adaptation strategy that takes into account energy consumption. We have considered two cases with the following combination of parameters: 1) SF7-TP14 and 2) SF8-TP8 because they consume similar energy as given in Table III [18]. Increasing TP from 8 dBm to 14 dBm roughly doubles the supply current and changing SF from 7 to 8 doubles the airtime. We expected that SF7-TP14 results in better performance because according to the SX1276 datasheet, passing from SF7 to SF8 increases the link budget by 2.5 dB while the difference in transmission power is 6 dB.

GW ID	**1**	**2**	**3**	**4**	**5**	**9**
Distance [km]	0.13	1.06	1.12	1.86	2.02	13.19
SF8-TP8 [%]	**95**	**94**	**84**	21	30	41
SF7-TP14 [%]	91	80	80	**44**	**42**	**63**

Table IV: PDR for two cases with similar energy consumption

Table IV shows measured PDR for both cases. We can observe an interesting phenomenon: SF7-TP14 obtains better PDR only for longer distances, while SF8-TP8 performs better for closer gateways. There is a need for more experiments to understand the reason of this effect.

VI. Conclusions and Future Work

This paper reports on the design and development of WiSH-WalT, a framework for controllable and reproducible LoRa testbeds. It takes advantage of two systems: WiSHFUL for the UPI interface that provides a flexible way for configuring the parameters of LoRa motes and WalT for easy deployment and monitoring of their operation. WiSH-WalT enables researchers and network architects to test their setup under real world conditions before final deployment and optimize the operation of LoRa motes with suitable parameters.

We present some experiments done on WiSH-WalT for evaluating the communication performance of a LoRa mote with several gateways. The measurements confirm that the performance behavior of LoRa motes follows a complex pattern based on multiple parameters. The presented results also shows that we need to perform more experiments to understand better all performance aspects. WiSH-WalT provides a suitable platform for doing this kind of research.

The current algorithm for bit rate adaptation requires a history of 20 packets to compute the adapted values of TP and SF. However, motes operate under low duty cycles and send packets infrequently, which may result in long adaptation times. Moreover, the algorithm only depends on SNR and our measurements show more complex dependency of PDR on communication conditions. Measurements with WiSH-WalT can provide an insight on whether it is beneficial to use ADR or disable it, which can reduce downlink traffic from gateways as well as save energy of motes.

In the future work, we plan to explore different approaches to bit rate adaptation and study the behavior of dense deployments in which contention may become important.

Acknowledgments

This work has been partially supported by the WiSHFUL European project No. 645274, the Use IoT project funded by Fonds Européens de Développement Régional (FEDER) No. N00011459 of the Auvergne-Rhône-Alpes region, and the French Ministry of Research project LabEx PERSYVAL-Lab (ANR-11-LABX-0025-01).

References

[1] D. Raychaudhuri, M. Ott, and I. Seskar, "ORBIT Radio Grid Tested for Evaluation of Next-Generation Wireless Network Protocols," in *Proc. of TRIDENTCOM*, 2005.

[2] C. Adjih, E. Baccelli, E. Fleury, G. Harter, N. Mitton, T. Noel, R. Pissard-Gibollet, F. Saint-Marcel, G. Schreiner, J. Vandaele, and T. Watteyne, "FIT IoT-LAB: A Large Scale Open Experimental IoT Testbed," in *Proc. of IEEE World Forum on Internet of Things*, 2015.

[3] P. Brunisholz, E. Duble, F. Rousseau, and A. Duda, "WalT: A Reproducible Testbed for Reproducible Network Experiments," in *IEEE INFOCOM International Workshop on Computer and Networking Experimental Research Using Testbeds (CNERT)*, San Francisco, USA, Apr. 2016, cNERT 2016 Best Demo Award. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01287566

[4] "Wishful: Wireless software and hardware platforms for flexible and unified radio and network control," 2015. [Online]. Available: http://www.wishful-project.eu

[5] K. Mikhaylov, J. Petäjäjärvi, and T. Hänninen, "Analysis of Capacity and Scalability of the LoRa Low Power Wide Area Network Technology," in *European Wireless 2016*, May 2016, pp. 1–6.

[6] O. Georgiou and U. Raza, "Low Power Wide Area Network Analysis: Can LoRa Scale?" *IEEE Wireless Commun. Letters*, vol. 6, no. 2, pp. 162–165, 2017.

[7] B. Reynders, W. Meert, and S. Pollin, "Range and Coexistence Analysis of Long Range Unlicensed Communication," in *IEEE ICT*, Thessaloniki, Greece, 2016, pp. 1–6.

[8] M. Bor, J. Vidler, and U. Roedig, "LoRa for the Internet of Things," in *EWSN '16*, Graz, Austria, 2016, pp. 361–366.

[9] A. Augustin, J. Yi, T. H. Clausen, and W. M. Townsley, "A Study of LoRa: Long Range & Low Power Networks for the Internet of Things," *Sensors*, vol. 16, no. 9, p. 1466, 2016.

[10] F. Adelantado, X. Vilajosana, P. Tuset-Peiró, B. Martínez, J. Melià-Seguí, and T. Watteyne, "Understanding the Limits of LoRaWAN," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 34–40, 2017.

[11] J. Haxhibeqiri, F. V. D. Abeele, I. Moerman, and J. Hoebeke, "LoRa Scalability: A Simulation Model Based on Interference Measurements," *Sensors*, vol. 17, no. 6, p. 1193, 2017.

[12] T.-H. To and A. Duda, "Simulation of LoRa in NS-3: Improving LoRa Performance with CSMA," in *IEEE ICC*, Kansas City, USA.

[13] T. Petrić, M. Goessens, L. Nuaymi, L. Toutain, and A. Pelov, "Measurements, Performance and Analysis of LoRa FABIAN, a Real-World Implementation of LPWAN," in *IEEE PIMRC*, Sept 2016, pp. 1–7.

[14] M. C. Bor and U. Roedig, "LoRa Transmission Parameter Selection," in *ACM DCOSS*, Ottawa, ON, Canada, 2017, pp. 27–34.

[15] A. Dongare, C. Hesling, K. Bhatia, A. Balanuta, R. L. Pereira, B. Iannucci, and A. Rowe, "OpenChirp: A Low-Power Wide-Area Networking Architecture," in *IEEE PerCom Workshops*, Kona, Big Island, USA.

[16] LoRa^{TM} Alliance, "A Technical Overview of LoRa and LoRaWAN." [Online]. Available: {https://www.lora-alliance.org/-portals/0/documents/whitepapers/LoRaWAN101.pdf}

[17] N. Sornin, M. Luis, T. Eirich, T. Kramp, and O.Hersent, "LoRaWAN Specification v1.0.2," 2016. [Online]. Available: https://www.lora-alliance.org/portals/0/specs/

[18] Semtech, "SX1272/73 - 860 MHz to 1020 MHz Low Power Long Range Transceiver," 2017. [Online]. Available: https://www.semtech.com/uploads/documents/sx1272.pdf

[19] M. N. Ochoa, A. Guizar, M. Maman, and A. Duda, "Evaluating LoRa Energy Efficiency for Adaptive Networks: From Star to Mesh Topologies," in *IEEE WiMob 2017*, Rome, Italy, Oct. 2017.

[20] W. Giezeman and J. Stokking, "The Things Network." [Online]. Available: https://www.thethingsnetwork.org