# Self-Configurable Wireless Automatic Irrigation System

César Penha*, Rogerio Campos-Rebelo * † ‡, and João Paulo Barros *†
*Polytechnic Institute of Beja, School of Technology and Management, Beja, Portugal
†UNINOVA, Center of Technology and Systems, Monte de Caparica, Portugal
‡NOVA University of Lisbon, Faculty of Science and Technology, Monte de Caparica, Portugal
cesar.penha@sapo.pt, rcr@uninova.pt, jpb@uninova.pt

***Abstract*—Agriculture in general and landscaping in particular remain as activities in which the amount of wasted water persists at very high levels. Despite all the technology available today, it is still common to find parks and green spaces where the applied irrigation systems are frequently triggered with dubious need. Currently, most irrigation systems are operated manually or are based on pre-defined controlled systems. Internet of Things, can help in information gathering, decision making, obtaining data through sensing, and giving feedback. This article presents an approach to develop a self-reconfigurable wireless automatic irrigation system. Two main contributions are proposed: (1) the development of the prototype of a wireless automatic irrigation system, supporting the automatic verification and selection of sensor devices, based on the required type of sensor and its respective received signal strength indicator and (2) the use of a graphical modelling formalism for the definition of rules that specify the desired irrigation behavior, allowing the automatic generation of code for microcontrollers. The system detects various environmental and soil parameters to optimize water consumption and automate irrigation control at a reduced cost. A proof of concept prototype is presented.**

***Index Terms*—Agriculture, Landscaping, Automatic irrigation, Internet of Things, Petri Nets, Model-Driven Development**

## I. INTRODUCTION

Water is an increasingly scarce resource [1] and its rational use is more than ever vital. Problems related to lack of water will increase if long-term forecasts on global climate change are confirmed. The meteorological records suggest significant increases in average temperature and decreases in annual precipitation, which will imply the reduction of available water resources in the 21st century [2]. Industry and tourism, among other productive activities, compete for this resource in order to increase their profitability and productivity. Adding to consumption, the waste in the use of water is another fact. At the national level, the sector where most water is wasted is agriculture [3]. In addition to this problem, and considering the expected increase in the world population, there is an urgent need to find solutions to ensure sufficient water supply for the entire population and avoid wastage. This can be achieved by increasing irrigation efficiency, incorporating practices and saving habits, and above all a greater awareness on this subject. One of the approaches to optimize the irrigation of agricultural fields and green spaces is the use of automatic irrigation controllers. Process automation has been applied in virtually every field of engineering with enormous success [4], although its implementation in agriculture, and in a particular way in precision irrigation, is somewhat limited. The key idea behind automatic control is the use of feedback [5]. In the field of autonomous irrigation, measurements of soil, plant, and climatic variables related to crop water requirements can provide information on the consequences of previous actions to calculate the next irrigation. Nowadays, in the era of advanced electronics and the Internet of Things (IoT) world, there are easily available options to counter the water waste.

Microcontrollers are everywhere, and at increasingly lower costs, development platforms proliferate, and wireless protocols facilitate communications. In this paper, we introduce the idea of developing a irrigation system dubbed SWAIS – Self-Configurable Wireless Automatic Irrigation System. An intelligent irrigation system has to have all the components that make the monitoring and autonomous control of the amount of water available to the plants without any failure or human intervention [6]. The system was developed to monitor certain environmental parameters, using various sensor devices spread across the green space in question. Then, the collected values will be sent using the LoRa (Long Range) [7] communication protocol to a central node to which a solenoid valve will be coupled, which will activate the watering. Presently, there are other communication protocols suitable for communication between IoT devices based on LPWAN networks, e.g. Narrow Band-IoT (NB-IoT) [8]. In this work, LoRa technology was chosen, mainly due to the following reasons:

- The prototype deployment location will be equipped with a LoRa infrastructure;
- The cost per device is lower with LoRa;
- LoRa devices have a higher latency. Latency is the time delay in transferring data after making a transfer request. A higher latency device ”checks in” with the network less often, increasing the battery life.

### A. Challenges

The development of this project involves several technical challenges. The system, as the acronym itself indicates, should

978-1-5386-9281-3/19/$31.00 ©2019 IEEE

2019 International Young Engineers Forum (YEF-ECE)
Caparica, Portugal, May 10, 2019

be self-configurable, thus allowing management of the type and quantity of sensors to use in a given park or green space. The definition of the rule or set of rules that allows activation of the watering should be user friendly. To that end, they were designed in a graphical environment using graphical modeling, allowing a user with no programming knowledge to change them. In the end, a functional prototype of the system, with all the necessary hardware, will be developed. This will include the programming of the microcontrollers associated with the their design and installation in field in order to be able to submit the prototype to a battery of tests. The main challenges raised in the development of this project were:

- Development of a redundant system at the level of installed sensors, so that in case of failure of one of the sensors, the system allows the use of another sensor of the same type within reach;
- The system must allow the use of external sensors, as far as the format of the transmitted payload respects certain requirements of the system itself;
- Implementation of hardware and software development so that energy consumption is reduced to a minimum, increasing the life of the batteries;
- Use of long range and low power wireless technology in communication between devices, namely LoRaWAN (Long Range Wide Area Network);
- The system will be autonomous at the energy level, as it will use batteries and solar panels, increasing its portability and the ease of adapting to existing infrastructures;
- The rules that define irrigation will be developed using Petri nets, namely the IOPT (Input-Output Place-Transition Tool) [9], using the automatic C code generator together with additional one.

## II. Background

Often we find irrigation systems that require an operator to intervene, namely opening and closing valves manually. Other times we find irrigation systems based on scheduling, which means that even in rainy conditions, they are activated for a certain period of time, often leading to water waste.

### A. Conventional irrigation systems

The conventional irrigation systems that we find in the green spaces, are composed mostly by sprinklers or drip irrigation hoses. These systems automatically or manually controlled do not have any feedback regarding the soil conditions, environmental conditions or even the type of plant or crop in a given location. Water is supplied to plants or crops in an almost ”random” way with parameters based on time or the gardener’s own will.

### B. Problems of conventional irrigation systems

In the case of manual irrigation systems, plants are often under stress due to large variations in soil moisture. This stress often leads to poor plant development and sometimes to death in the case of more sensitive plants or crops [10]. The absence of an automatism leads to under-irrigation or over-irrigation of soil. As already mentioned, currently the problem of water waste is a topic of great importance and object of several studies and works. In simpler automatic or automated watering systems, this problem is partially solved. These systems are mostly based on watering controllers that are programmed as a function of time, that is, they always apply the same amount of water in a certain space. The automation releases the manual work of opening and closing irrigation valves, and provides a more or less constant watering. However does not solve all problems. [11] Most of these irrigation systems do not have any feedback regarding environmental or soil parameters. The key idea behind automatic control is the use of feedback. In the field of autonomous irrigation, measurements of soil, plant and climatic variables related to crop water requirements can provide information on the consequences of previous actions to calculate the next irrigation.

## III. System Architecture

The SWAIS consists of a sensor or several sensor devices and an actuator device. A sensor device is constituted by a microcontroller, a sensor and a communication module. The sensor may be of various types eg. sensor for soil moisture, temperature, rain, etc. The actuator device consists of a communication module, a microcontroller and an actuator (irrigation valve). Its architecture is shown in Fig. 1.

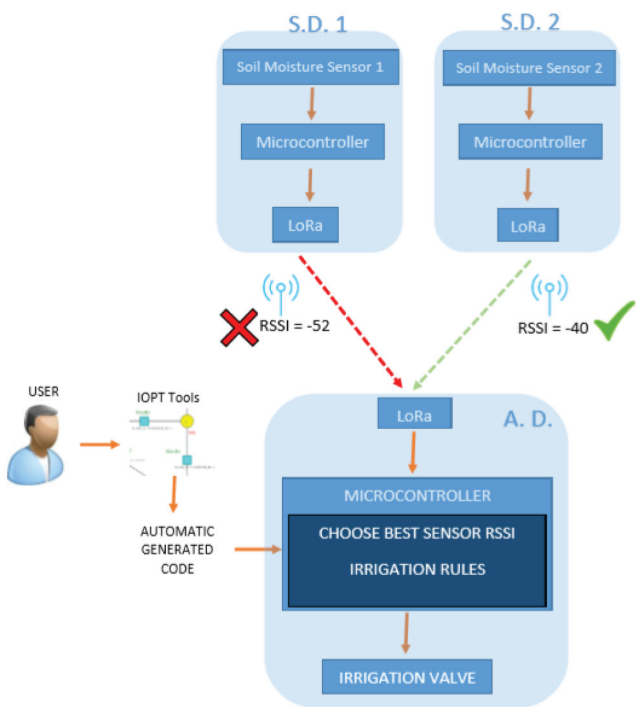


Fig. 1. SWAIS Prototype architecture. We can see two sensor devices (S.D.) of the same type (soil moisture) and one actuator device (A.D.).The sensor of the same type with the best RSSI is the chosen one.

The system is receiving continuous feedback from the sensor devices scattered around the landscape. The actuator device receives the data from the sensing device(s) and checks the power of the signals individually by selecting the best RSSI (Received Signal Strength Indicator) from all devices of the same type within its reach. The user graphically develops a graphical model using the IOPT Tools framework[1] [9], where the irrigation rules are specified. The code generated automatically by the tool is combined with the developed code to obtain the best RSSI and uploaded to the microcontroller. The microcontroller acts on the solenoid valve responsible for landscape irrigation. Alternatively, the system may receive data from other non-SWAIS sensors such as street sensors or weather stations. For this to be possible, certain parameters of the communication between these sensors and SWAIS must be taken into account. When using LoRa communication modules to receive the data, the string referring to the transmitted payload must have the same format so that it can be recognized by the actuator module. In addition to this requirement, the external sensors must communicate in the same frequency band and for the same internal network. In SWAIS, these and other parameters are defined in the LoRa communication modules and can be changed as required on the manually developed code. If we use LoRa gateways to receive the data from sensor devices these requirements are unnecessary since gateways operate in several frequencies and receive any string format.

## IV. Automatic Signal Verification

As already mentioned, SWAIS consists of several sensor devices and an actuator device. In Fig.2 we can observe an implementation plan of SWAIS in Carlos Boto park in Lagoa (Algarve, Portugal), where the actuator device is in a central area of the park, next to a water point of an existing infrastructure. In this case, a range of the communication module in the order of 250 meters was considered, which is well below the reference range of the communication modules to be implemented. Within this range represented by the red circle, the actuator device will select which sensors are required for the feedback for which it is programmed, and within which they present the best RSSI which, in principle, will be closest to the actuator device. We can see that within the scope there are three soil moisture sensing devices, and a rain sensor device, but in fact only two will be used; one of soil moisture and one of rain. At first glance it seems unnecessary to have more than one sensor of the same type in the same space, but with a closer look, we can see some advantages:

- In the case of plants or crops of different types and, consequently, with varied water needs, it may be necessary to use sensors of the same type located in the different areas where those crops are, segmenting the irrigation area according to the needs;

[1]IOPT-Tools site: http://gres.uninova.pt/IOPT-Tools

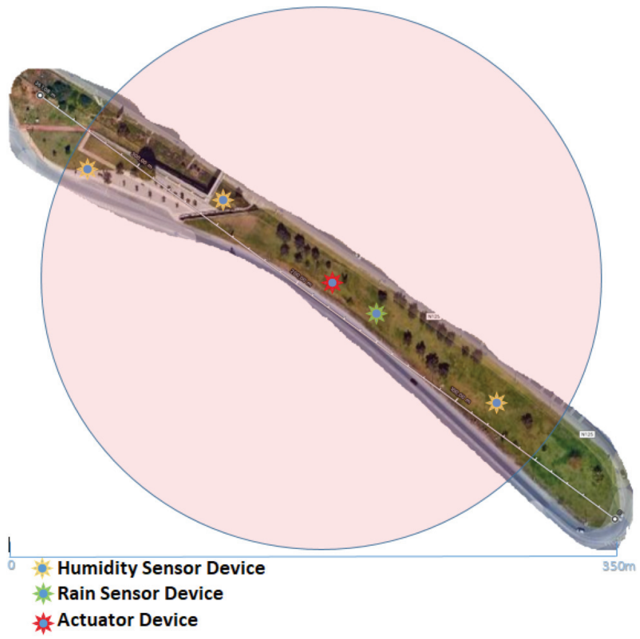


Fig. 2. SWAIS Prototype implementation example in Carlos Boto park in Lagoa. The red circle represents the range of the actuator device, and within it the several sensor devices.

- In case of failure of one of the sensors, SWAIS can use another sensor of the same type that is within reach; this allows redundancy in the system, therefore, it is better to have a sensor that is a little further away, or even one that does not belong to the same park or landscape, than to have no feedback parameter.

The verification of the sensor devices is performed when the system is switched on for the first time; all devices with the best RSSI are stored in an array. From there, only the saved sensor devices will be used. However, the user can decide on the time until a new scan for new devices, or check if the RSSI strength between devices stored in the previous scan is retained. The SWAIS system supports a network between sensor devices and actuating devices independent of the rest of the existing infrastructure (e.g., LoRa gateways). However, for each coupled sensor device, it is possible for each actuator device to send data to a gateway external to the network.

When we extrapolate SWAIS to a small city, we can observe in Fig.3 that the overlap of the communication modules reach is considerable. As an example, several parks were selected in the city of Lagoa where there is a need for irrigation (automatic or manual). An average range of 250 meters has been estimated for each device, which by default is well below that announced by the manufacturer of the communication module. Tests have proved that the LoRa module used in this prototype achieves distances of about 15Km in Line of Sight (LoS), depending on the emission power programmed in the module itself. It is easy to see the redundancy of devices and the coverage of the city by the actuating devices.

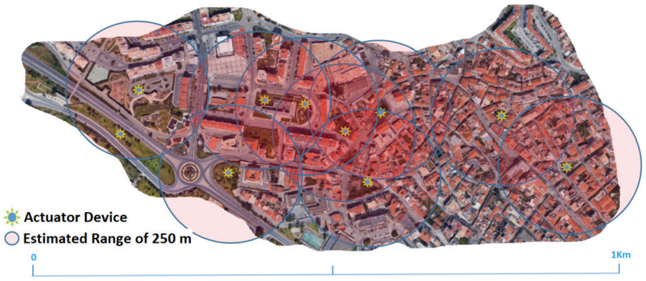


Fig. 3. Lagoa city (Algarve) SWAIS coverage example: Each actuator device is in the center of a circle.

## V. Irrigation rules model

Petri nets are a language for modeling and developing systems using a precise and graphical representation of the system, often considered an extension of the finite state machines. The analysis of a Petri net model allows the evaluation of the dynamic behavior of the modeled system. The result of this evaluation may lead to improvements or changes to the system [12]. Considering the aforementioned, and in order to facilitate the development of the part of the project referring to the rules that culminate in the activation or not of irrigation, the IOPT Tools, a web based tool framework, was used in order to engage the development of project SWAIS. This IDE offers a complete set of development and testing tools for microcontrollers, industrial automation applications and other digital systems. The tool is freely available online in a graphical web-based interface. IOPT-Nets [9] are a class of non-autonomous Petri nets because they allow explicit modeling of inputs and outputs in the net model, allowing the reading of sensors, actuation on other devices, and communication with other systems [13]. The tool consists of an interactive graphical Petri net editor [14] [15] for designing IOPT models, a simulator [16] [17]to test system behavior, a state-space generator [18] [19], and a query system to automate model checking and property checking. The advantages of using this set of tools are diverse. We underline the following:

- It allows the early detection of errors in the design, allowing correction of most errors before reaching the implementation phase;
- Automatic code generation eliminates the need to write code (C, VHDL, JavaScript) manually, except for some device-specific operations or assignment of input and output pins;
- Minimization of errors during the implementation phase due to low-level coding [20].

### A. *Verification of irrigation rules*

An example of a model for the irrigation rules is presented in Fig. 4. In this case, only two sensors were used (rain and soil moisture), but any additional number of additional ones could be present. The base network provided to the user will be the one shown outside the green and red rectangles. The red rectangle represents the model part corresponding to the conditions to initiate irrigation. The green rectangle represents the model part corresponding to the conditions to stop irrigation. The user can create the model in the way that best suits some specific irrigation requirements.

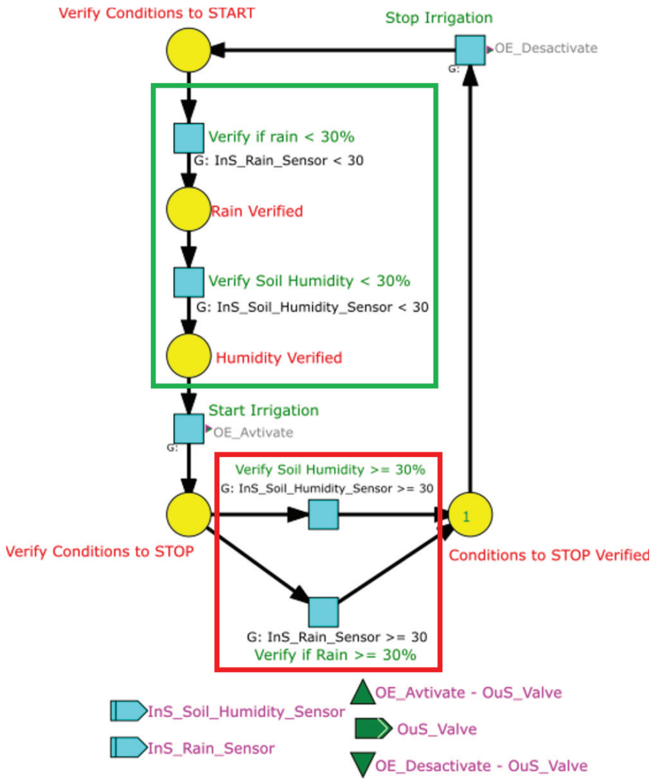


Fig. 4. SWAIS Petri Net schematic developed with IOPT Tools framework, represents the rules for irrigation. In this case were used two sensors ( rain and soil moisture).

The prototype uses analog sensors whose output is a voltage value converted into a 10-bit scale (0 to 1023). The value is mapped to a scale between 0 and 100%. In the case of the rain sensor, when rain drops begin to fall on the sensor the value increases. Regarding the humidity sensor, the principle is the same, i.e., the higher the humidity in the soil, the higher the value of the sensor.

The verification of the defined rules start with the input values collected by the sensors in the network. Then, the controller checks if the rain sensor has a value less than 30% (range 0% to 100%). If a value less than 30% is read, the program will carry out the next check, which in the presented case will be a soil humidity sensor. If this sensor value is less than 30% (range 0% to 100%) the controller starts watering (output event in transition "Start Irrigation"). When the rain sensor value is equal or higher than 30%, the system will generate an alert and will stop watering, not performing the second check (soil humidity sensor) thus closing the program. The same is true if the value of the soil moisture sensor is equal or higher than 30%.

The timing of the program loop will be defined by the user. He can add a ”delay” to slow up the program, i.e., activate the watering for five minutes and, after that, verify the sensor values again. In all cases where the irrigation is not activated by any of the sensors that exceed the defined maximum values, the system will emit an alert output by means of a LED (Light-Emitting Diode). For simplification purposes, the actuation of this alert LED is not yet modeled in the Petri net for the prototype of Fig. 4.

Finally, there is also a functionality called ”manual override” that allows the user to start or end the watering regardless of sensor values. This functionality is activated by a button located on the actuator device.

## VI. Code Integration

The IOPT Tools supports the generation of ANSI C code allowing in this way the generation of code for a wide range of platforms [12]. The code generated by the tool was adapted for the purpose of using it in the Arduino development platform where the prototype was developed, e.g. renaming files generated with the .c extension for files with the .ino extension so that they can be editable by the IDE of the Arduino platform. Some libraries were included, as well as some global variables necessary for the proper functioning of the program. Manually developed code has the following features:

1) The choice of the type of sensors (Rain, soil moisture, temperature, relative humidity, soil temperature, luminosity, etc.) through the needs that the user deems convenient for the proper development of a particular crop. In other words, the user may find it necessary that a certain green space needs only one type of sensor, while any other space requires two or more types of sensors in order to adjust the irrigation. In this way, and although there may be several types of sensors within reach, the system chooses only the sensors for which it has been programmed, discarding the remaining sensors.
2) The choice of sensors for better RSSI, that is, having two or more sensors of the same type within reach, the system will choose the one that has the best RSSI. This functionality allows redundancy between sensors in the event of signal failure or malfunction, since neighboring sensors can always be used within range.

The code generated by the tool has been slightly adapted and interconnected with the manually performed code regarding the verification part and the choice of the devices with the best RSSI. Subsequently it will be introduced in a microcontroller (Arduino) in its Nano version, which controls the actuating device.

## VII. Prototype Implementation

The whole prototype was developed considering its low final cost and its energy independence, using batteries and solar panels, there is no need for a connection to the power grid . The components needed to build the prototype are easy to acquire - easily purchased on the Internet - and inexpensive. Fritzing software [2] was used to make the connections between components and to draw the printed circuit board where the components will be soldered. The prototype is divided into two main components: sensor devices and actuating devices.

### A. Sensor devices hardware

The sensor device is able to transmit data to the actuator device using LoRa technology. The coupled sensor can be of various types such as soil moisture, rainfall, ambient temperature, soil temperature, luminousness, among others. In each sensor device only one type of sensor is connected, and this is chosen according to the needs of the place to be irrigated and the needs that the user deems necessary. Several sensor devices may coexist in one location, each with a different sensor type. Each sensor device consists of a sensor, a microcontroller, a LoRa module, and a battery. The microcontroller is an ATiny85 [21], a high-performance, low-power Microchip 8-bit AVR RISC-based microcontroller which combines 8KB ISP flash memory, 512B EEPROM, 512-Byte SRAM, 4-channel 10-bit A/D converter, and 6 general purpose I/O lines. It has three software selectable power saving modes, and debugWIRE for on-chip debugging. The device achieves a throughput of 20 MIPS at 20 MHz and operates between 2.7-5.5 volts. The choice of integrated LoRa communication, fell on the chip SX1276 from Semtech used by REYAX in its RYLR896 module [22]. The RYLR896 transceiver module features the Lora long range modem that provides ultra-long range spread spectrum communication and high interference immunity while minimizing current consumption. The configuration of the module is performed through AT commands, which facilitates the task of changing the various configuration parameters. The operating voltage is between the 2 and 3.6 volts, and its consumption can vary between 43mA in transmission mode and 0.5 uA in sleep mode. According to the manufacturer, communication reach can be up to 15Km, depending on the configuration of the module itself. The module is certified by NCC (National Communications Commission) and FCC (Federal Communications Commission). The power supply to the sensor device may be two rechargeable or non-rechargeable AA batteries or a 3.7-volt 1-cell LiPo battery. The sensor device is equipped with a two in one Buck-Boost DC to DC converter, which converts input voltages between 0.9 and 6 volts to a regulated output voltage of 5 volts.

### B. Actuating devices hardware

The actuator device is responsible for receiving the data from the sensor device(s), choosing the identical devices with the best RSSI, checking the irrigation rules and determining whether it is necessary to irrigate by opening or closing the

[2]Fritzing site: http://fritzing.org/home/

irrigation valve. A LoRa communication module, similar in every way to the module used in the sensor devices, was used to receive the values of the sensors coming from the sensor devices. To activate or deactivate watering, a 1/2-inch, 4.5 volts solenoid valve with a 9 Ohm coil resistance and a working pressure of 0.02 to 1 Mega Pascal was used. Regarding the alarm system, there are two RGB LEDs: the first to monitor the battery charge in which the LED is green and the battery is charged and the blue LED when battery is charging. The second LED refers to communications where the green LED is on, a communication is being received, the blue LED is on, a communication is being send (in a future work), and the red LED is on, where there has been an error in the communication. To process all these steps, an Arduino microcontroller was used in its Nano version [23], equipped with an ATmega328P processor. The power circuit consists of a 6-volt solar panel capable of producing a maximum of 350 mAh, a charge controller of the company Adafruit [24] able to balance the energy received by the solar panel and the load. In order for the actuator device to work even at night, a 4000 mAh LiPo battery was used. To implement a test prototype, a box like the one observed in Fig. 5 was developed in order to house the electronics related to the sensor devices. This box was designed to accommodate various types of sensors, the LoRa communication module, the microcontroller and the battery. For this purpose, the box was designed using Freecad software and printed on a 3D printer so that it can be sealed and placed at ground level. The box may be partially or totally buried. If it is partially buried (up to the level of the spigot), it is possible to equip it with a small solar panel at its top, while at the same time measuring the temperature and humidity of the soil, as they are level of the spigot. If the box is completely buried, the system will only resort to the internal battery and the range of the communication module will be reduced.

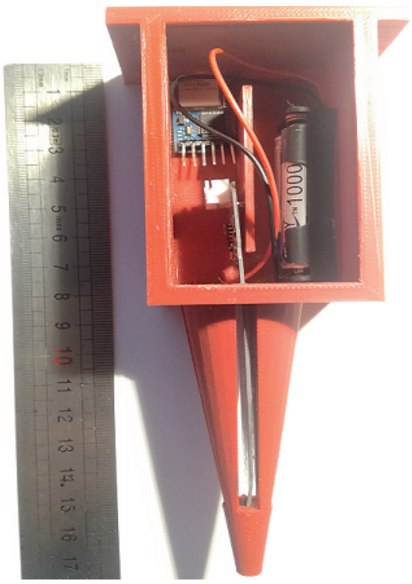

Fig. 5. SWAIS sensor device box with LoRa communications module, battery and soil moisture sensor in the spigot.

## VIII. Conclusions

In this article, the operation of conventional irrigation systems and the their most common problems were explained, namely regarding the lack of feedback. A solution for self reconfiguration of the sensors scattered around the landscape was also demonstrated.

In an initial phase, the configuration process is based on the selection of one or several types of sensors considered necessary for the best growth of the existing culture in that landscape. The choice is made considering the available sensors and the required reach of the actuator device. As a second step, the best sensors of the same type are selected (if the is more than one in range), based on the power of the radio signal received by the actuator device. An implementation of SWAIS in a particular park, and another one in a small city, demonstrated a possible coverage of the prototype if an actuator device was implemented in each park of the city. It is easy to observe the redundancy that can be obtained in the event of communication failure or failure of one or mere sensors in a certain space.

The system also allows the use of external sensors such as weather stations and street sensors, provided the payload format is taken into account, so that it can be recognized by the actuator device and allow irrigation. Another contribution of this work was the implementation of irrigation rules using Petri nets. It is intended that the modeling of the rules become accessible to any user who does not have programming knowledge. The IOPT Tools allow an easy graphical presentation of the rules that will determine the irrigation, generating the code automatically.

Finally, one implementation of the prototype was demonstrated, taking into account the necessary hardware components and the electronic schematics of the sensing devices and the actuating devices. We can conclude that the integration of Internet of Things with irrigation systems can allow the intelligent automation of these systems allowing to reduce water consumption and waste. The adoption of these technologies incorporates new benefits into decision support systems, monitoring and water management, while also having energy concerns.

## IX. Future work

Future work will take into account the implementation of SWAIS in public parks and other terrain in order to carry out large-scale implementation tests and solve possible problems in the devices communication and installation. The actuator device will have an additional functionality to transmit the data received, by the communication module and the sensors that it is using, to a gateway. In this way the data will be available in the Internet for later monitoring and control. Another aspect that will be taken into account in future work will be the development of a script that will allow to automatically join the manually developed code with the code generated by the

IOPT Tools. In this way, the user will only have to worry about the network design as there will be no need to code.

## Acknowledgment

This work was partially financed by Portuguese Agency FCT – Fundação para a Ciência e Tecnologia, in the framework of project UID/EEA/00066/2019.

## References

[1] K. Bellware. Global Water Shortage Risk Is Worse Than Scientists Thought. Accessed on 2019-02-08. [Online]. Available: https://www.huffingtonpost.com/entry/water-scarcity-study_us_56c1ebc5e4b0b40245c72f5e?guccounter=1

[2] J. B. Hugh Turral and J.-M. Faurès, *Climate change, water and food security*. Food and Agriculture Organization of the United Nations, 2016, http://www.fao.org/3/i2096e/i2096e00.htm, accessed 2019-02-08.

[3] A. Cristas. Programa Nacional para o uso eficiente da água. Accessed on 2019-02-08. [Online]. Available: http://www.apambiente.pt/_zdata/CONSULTA_PUBLICA/2012/PNUEA/Implementacao-PNUEA_2012-2020_JUNHO.pdf

[4] S. Bennett, "A brief history of automatic control," *IEEE Control Systems Magazine*, vol. 16, no. 3, pp. 17–25, 1996.

[5] A. Mccarthy, N. Hancock, and S. Raine, "Advanced process control of irrigation: The current state and an analysis to aid future development," *Irrigation Science*, vol. 31, 05 2011.

[6] R. Ferrarezi, G. Weaver, M. Van Iersel, and R. Testezlaf, "Subirrigation: Historical overview, challenges, and future prospects," *HortTechnology*, vol. 25, pp. 262–276, 06 2015.

[7] LoRa Alliance. LoRa. Accessed on 2019-02-24. [Online]. Available: https://lora-alliance.org/

[8] 3GPP. NarrowBand IOT. Accessed on 2019-04-27. [Online]. Available: https://www.3gpp.org/news-events/1733-niot

[9] L. Gomes and J. P. Barros, "Refining IOPT Petri nets class for embedded system controller modeling," in *IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*, Oct 2018, pp. 4720–4725.

[10] A. Berg, J. Byrne, and R. Rogerson, "An urban water balance study, lethbridge, alberta: Estimation of urban lawn overwatering and potential effects on local water tables," *Canadian Water Resources Journal / Revue canadienne des ressources hydriques*, vol. 21, no. 4, pp. 355–365, 1996. [Online]. Available: https://doi.org/10.4296/cwrj2104355

[16] GRES Research Group. IOPT Tools Simulator. Accessed on 2019-03-12. [Online]. Available: http://gres.uninova.pt/IOPT-Tools/simulator.php

[11] N. Durga and M. Ramakrishna, "Smart irrigation system based on soil moisture using iot," *International Research Journal of Engineering and Technology*, vol. 05, no. 06, pp. 2003–2007, 2018.

[12] R. Campos-Rebelo, F. Pereira, F. Moutinho, and L. Gomes, "From IOPT Petri nets to C: An automatic code generator tool," in *2011 9th IEEE International Conference on Industrial Informatics*, July 2011, pp. 390–395.

[13] L. Gomes, J. P. Barros, A. Costa, and R. Nunes, "The input-output place-transition petri net class and associated tools," in *2007 5th IEEE International Conference on Industrial Informatics*, vol. 1, June 2007, pp. 509–514.

[14] GRES Research Group. IOPT Tools Editor. Accessed on 2019-03-12. [Online]. Available: http://gres.uninova.pt/IOPT-Tools/pnml_editor.php

[15] F. Pereira, F. Moutinho, J. Ribeiro, and L. Gomes, "Web based IOPT Petri net editor with an extensible plugin architecture to support generic net operations," in *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society*, Oct 2012, pp. 6151–6156.

[17] F. Pereira and L. Gomes, "Cloud based IOPT Petri net simulator to test and debug embedded system controllers," in *Technological Innovation for Cloud-Based Engineering Systems*, T. A. D. O. G. Camarinha-Matos, Luis M.and Baldissera and F. Marques, Eds. Cham: Springer International Publishing, 2015, pp. 165–175.

[18] GRES Research Group. IOPT Tools Editor. Accessed on 2019-03-12. [Online]. Available: http://gres.uninova.pt/IOPT-Tools/ss_init_marking.php

[19] F. Pereira, F. Moutinho, L. Gomes, J. Ribeiro, and R. Campos-Rebelo, "An IOPT-net state-space generator tool," in *2011 9th IEEE International Conference on Industrial Informatics*, July 2011, pp. 383–389.

[20] L. Gomes, F. Moutinho, and F. Pereira, "IOPT-tools – a web based tool framework for embedded systems controller development using Petri nets," in *2013 23rd International Conference on Field programmable Logic and Applications*, Sep. 2013, pp. 1–1.

[21] Atmel. Atmel 8-bit AVR Microcontroller with 2/4/8K Bytes In-System Programmable Flash. Accessed on 2019-02-18. [Online]. Available: https://www.microchip.com

[22] R. T. Corporation. REYAX RYLR896 . Accessed on 2019-02-18. [Online]. Available: http://reyax.com/products/rylr896/

[23] Arduino. Arduino Nano. Accessed on 2019-02-18. [Online]. Available: https://store.arduino.cc/arduino-nano

[24] Adafruit. Adafruit USB, DC & Solar Lipoly Charger. Accessed on 2019-02-18. [Online]. Available: https://learn.adafruit.com/usb-dc-and-solar-lipoly-charger/overview