Expert Systems With Applications 199 (2022) 116974

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

IoT platform for failure management in water transmission systems

José Pérez-Padillo *, Francisco Puig, Jorge García Morillo, Pilar Montesinos

Department of Agronomy, University of Córdoba (Hydraulic Engineering Area), Campus de Rabanales, Ed. Da Vinci, 14071 Cordoba, Spain

ARTICLE INFO

Keywords:
Decision support system
Wireless sensors
Artificial intelligence
Web services
Information and communication technologies (ICT)
Digitization

ABSTRACT

In aging water supply systems, many components have exceeded their service life. Consequently, tools are needed to efficiently manage the failures in these systems. This work describes the development and implementation of a web tool for the management of breakdowns in water transmission networks. The proposed tool, called wAIter, consists of a network of wireless water pressure sensors that send real-time data to an IoT platform. The core of the platform consists of a rule-based decision algorithm, which detects and classifies failures based on the recorded pressure values and then sends an alert to repair them. In addition, wAIter uses the mathematical model of the hydraulic network to estimate the maximum repair time without causing supply interruptions. This information is key in the decision-making process to repair breakdowns and facilitate repair work management. Finally, the results of the implementation in a real water transmission network are presented.

1. Introduction

The current growth of the world's population requires the careful management of available resources. The United Nations predicts that up to 6.5 billion people will live in urban areas by 2050. This upward trend in population density in large cities increases exposure to extreme weather events (Herrera, Ferreira, Coley, & De Aquino, 2016) and longer periods of drought are alerting researchers to the importance of good water resource management (Butler et al., 2014).

The age of many water distribution systems considerably affects their operation. These systems have been in service for more than 50 years, so a significant number of their components (pipes, valves and pumps) have exceeded their lifespan, in addition to being manufactured with obsolete materials. Consequently, the frequency of failures tends to increase over time (Winkler, Haltmeier, Kleidorfer, Rauch, & Tscheikner-Gratl, 2018). This has a negative impact on the performance of water supply networks which, together with the increasing trend of energy prices, makes the modernization of these infrastructures essential (Hernandez & Kenny, 2010).

The operation of water transmission networks (WTN) that convey water from sources to regulation tanks in municipalities is conditioned by their dispersion in the territory. These networks are made up of long pipes (tens of kilometers in length) located in non-urban areas, which makes it difficult to monitor all the key points of the network on site daily. In addition, given the large diameter of the pipes to adequately transport large flows, leaks in WTNs result in the loss of substantial volumes of water.

Numerous techniques have been developed to improve the performance of large water supply networks. In addition to the traditional techniques of periodic acoustic measurements (Cody, Tolson, & Orchard, 2020; Fuchs & Riehle, 1991) and minimum night flow analysis (McKenzie & Seago, 2005), there are techniques to analyze data from the SCADA data acquisition system in real time and detect leaks (Romano, Kapelan, & Savić, 2014; Wu, Sage, & Turtle, 2010) or compare real data to the data generated by hydraulic models (Al-Khomairi, 2008; Shao, Li, Zhang, Chu, & Liu, 2019). The optimal placement of sensors in hydraulic networks to maximize fault detection in the network has also been studied (Soldevila, Blesa, Tornil-Sin, Fernandez-Canti, & Puig, 2018).

However, these techniques have important limitations to implement them in real systems. On numerous occasions the technology has only been validated in the laboratory (Fereidooni, Tahayori, & Bahadori-

Abbreviations: API REST, Representational state transfer application programming interface; AWS, Amazon Web Services; GSM, Global System for Mobile Communications; HTTP, Hypertext Transfer Protocol; IoT, Internet of Things; PaaS, Platform as a Service; LoRaWAN, Long Range Wide Area Network; LPWAN, Low-power wide-area network; MDT, Mean detection time; MRT, Maximum repair time; NB-IoT, Narrow Band Internet of Things; SCADA, Supervisory control and data acquisition; SF, Safety factor; SPA, Single-age applications; SQL, Structured Query Language; WTN, Water transmission network; wAIter, Artificial Intelligence Water Platform.

* Corresponding author.
E-mail addresses: g22pepaj@uco.es (J. Pérez-Padillo), g32pupef@uco.es (F. Puig), jgmorillo@uco.es (J. García Morillo), pmontesinos@uco.es (P. Montesinos).

https://doi.org/10.1016/j.eswa.2022.116974
Received 17 May 2021; Received in revised form 1 December 2021; Accepted 22 March 2022
Available online 24 March 2022

Jahromi, 2020). The implementation costs are high, since they require a large number of sensors to ensure acceptable results. In addition, to apply current machine learning techniques, historical data series are needed at several points in the network.

The development of new telecommunication systems based on low power wide area networks (LPWANs) is driving the development of telemetry and data acquisition systems to collect large amounts of information (Lalle, Fourati, Fourati, & Barraca, 2019). Systems based on the Internet of Things (IoT) are enabling the decentralized monitoring of large hydraulic networks and providing access to information in real time from any point with an internet connection (Apostol, Truică, Pop, & Esposito, 2021; Narayanan, Sankaranarayanan, Rodrigues, & Lorenz, 2020). To efficiently manage these large hydraulic infrastructures, their digitization is essential. Digitization is the only way to apply Big Data techniques to the recorded data and facilitate the daily management tasks of this type of facilities, thus making them safer and more resilient to adverse situations (Makropoulos & Savic, 2019).

This paper describes the development and implementation of an IoT platform aimed at fault detection in drinking water transmission networks using open-source software. The core of the platform is a rule-based decision algorithm, which detects and classifies faults using only the pressure data recorded by the linked network of low-cost wireless pressure sensors. Upon detection of a fault, the system sends alerts and estimates the maximum repair time without causing supply outages, facilitating the management of repair works. The applicability of the proposed system has been tested in a real WTN.

The rest of this article is organized as follows. Section 2 describes the architecture of the IoT platform for WTNs. Section 3 explains the functions of the different modules of the platform. Section 4 presents a case study of the implementation of the platform. Section 5 discusses the results and Section 6 concludes.

1.1. IoT platform architecture

The architecture of the IoT platform proposed in this work is organized into three independents but connected layers (Fig. 1). The platform is called wAIter, a combination of the words “water” and “artificial intelligence”. The wAIter platform architecture allows adding, deleting, or updating the modules and components of the layers without affecting the system architecture. These layers are described below.

1.2. Layer 1: Data collection

Layer 1 contains the IoT sensors that record the required variables in WTNs. The sensors are controlled by Arduino-type microprocessors and send the information to the cloud using LPWAN technology. Each device has a unique identifier that allow the elements in the other layers of the platform to identify the sensor that captured each data.

1.2.1. Pressure sensor network

This level contains the sensors that transform the hydraulic variables involved in failure detection (e.g., pressure) into electrical signals. Ad-hoc communication nodes have been developed for this research (Pérez-Padillo, Morillo, Ramirez-Faz, Roldán, & Montesinos, 2020). These nodes can read different types of signals (4–20 mA, analogue signal, digital signal, I2C signal) and can also be powered in three ways: with batteries, with a small photovoltaic module or by connecting the device to the conventional electrical network (Fig. 2).

Each device is equipped to host an Arduino MKR family hardware

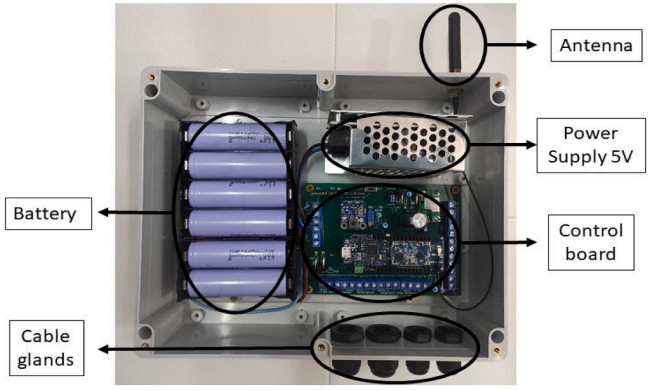

Fig. 2. Communication node components.

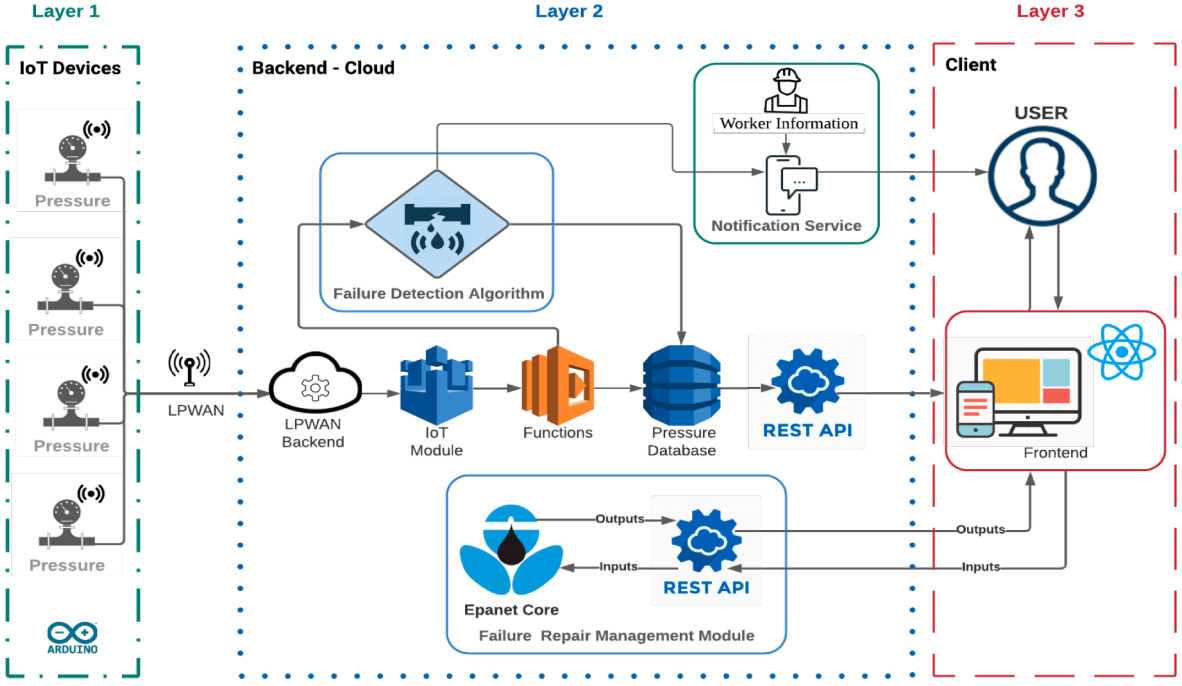

Fig. 1. Architecture of the pressure monitoring system.

board. These boards are composed of an Atmel SAMD21 microcontroller. All of them have the same input and output layout and are compatible with the rest of the electronics of the measurement device. The Arduino microcontroller collects hydraulic data at different periods and disconnects the system between measurements to save battery power. This helps to reduce the energy consumed in each reading cycle and increase the autonomy of the device.

The communication node acquires and periodically transmits the pressure data recorded by the pressure transducer. The pressure records are used to characterize the operation of the system. Thus, pressure values out of the normal range indicate system malfunctions.

1.2.2. Wireless communication network

The expansion of IoT systems is driving the development of LPWANs. Due to the range and power consumption of these networks, they are ideal for communicating scattered points in large and often difficult to access territories, where WTN monitoring systems are often located (Singh, Puluckul, Berkvens, & Weyn, 2020). The most widely used LPWAN communication networks are Sigfox (Purnama & Nashiruddin, 2020), LoRaWAN (Cesana & Redondi, 2017; Semtech Corporation. Lora Overview, n.d.) and NB-IoT (Chen et al., 2017), whose main features are shown in Table 1.

The selection of the most suitable communication technology depends on each application case. The IoT sensor described in the previous section is versatile, so it can communicate with any of the LPWANs in Table 1 using the corresponding microcontroller. The Arduino MKR family has different boards, each of which is adapted to a communication system: LoRa (Arduino MKR 1300), Sigfox (Arduino MKR 1200), Wifi (Arduino MKR 1000), GSM (Arduino MKR 1400), and Narrow Band IoT (Arduino MKR 1500).

1.3. Layer 2: Backend

The backend layer provides the platform services and contains the utilities for sensor control, data analysis and storage, notification, application programming interfaces that conform to the constraints of representational state transfer architectural style (REST API) allowing for interaction with REST web services (Kumar Polu, 2018), and the failure identification algorithm. The development of the backend in the cloud allows new services to be added without affecting the system architecture, as well as to scale and/or replicate the platform on other servers and connect it to any client or service with their own interface.

In this work, the Amazon Web Service (AWS) platform has been chosen to offer PaaS (Platform as a Service) type services, thus avoiding the need to manage the web infrastructure (hardware and operating systems). This feature facilitates platform development and implementation. The architecture has been developed to easily add or remove modules, thus ensuring the scalability of the system. The modules that form the backend are:

- LPWAN-Backend: the measurement devices are connected to the LPWAN cloud where the values arrive in hexadecimal format. The call-backs that redirect the sensor value through the Hypertext Transfer Protocol (HTTP) to the backend of the platform are configured from the LPWAN cloud. This system allows managing several devices together, which facilitates the process.
- IoT service: in addition to storing sensor data in the cloud, this element permits processing a variety of data in an easy manner. The service facilitates the management of large amounts of information, such as tools for developers, data security, administration tools, data analysis, etc.
- Functions: This element manages and redirects the data transferred from the LPWAN to the different cloud services: database and calculation algorithms. Each pressure value has an identifier associated with the sensor that registered it.
- Database: The DB Dynamo database (NoSQL type) stores both the data recorded by the sensors and the results of the calculation algorithms. This database was selected due to its ease of integration with other cloud services.
- Failure Detection Algorithm: This algorithm detects possible faults in the WTN and analyzes the pressure data to determine whether the values are within the normal operating range of the pipe. If the values are not within the normal range, it sends an alert command to the notification module which is also stored in the database. This algorithm is described in Section 3.1.
- Notification services: This module sends failure alerts to the platform users.
- Failure repair management module: The availability of the mathematical model of the network is required for using this module. It contains an ad-hoc algorithm written in Python programming language (Python Software Foundation. Python Language Reference. Version 3.6.11 Available at https://www.python.org, n.d.) that is connected to the EPANET open-source hydraulic simulator (Rossman, 2000) as a calculation engine to analyze the behavior of the system during different failure scenarios. The algorithm is hosted in a cloud server and accessed through a REST API created with the Django open-source framework (Django, 2020). The API was configured to receive HTTP calls through a POST method that sends the necessary parameters for the algorithm execution (Kumar Polu, 2018). A detailed description of this module is given in section 2.4.

1.4. Layer 3: Frontend

The frontend layer contains the platform's graphical interface for the user and establishes the connection between the database, the user, and the calculation algorithms. It enables users to operate the platform and visualize all its functionalities in a simple and efficient manner. The frontend interacts with the REST API (backend) by the HTTP protocol for the user to make requests for information to the database and the analysis algorithms hosted in the backend.

To develop the interface, the open source ReactJS library for creating single-page web applications (SPA) was used (Gackenheimer, 2015). This library allows the platform utilities to be divided into components, each with its own logic and independent operation.

Depending on their category, users can use the frontend functions described in section 3. The user categories are administrator, who has access to all the functions, and qualified and basic users, who are permitted different levels of accessibility to the platform functions. The access levels are given in the description of the functions.

The role of administrator is reserved for the platform developers, who are the only ones that can add and remove users and manage databases. Administrators have full access to the functions allowed to users in the other categories. WTN managers have the role of qualified users. They are enabled to create, remove sensors, and edit the pressure range

Table 1
Comparison of low power wide area networks.

Attribute	NB-IoT	LoRa	SigFox
Operation frequencies [MHz]	700–900	868	868
Band	Cellular, licensed	ISM, unlicensed	ISM, unlicensed
Transmit power [dBm]	23/35	14	14/27
Bandwidth [kHz]	180	125	0.1/0.6
Range (km)	<15	9 km (Urban) 50 km (Rural)	10 km (Urban) 50 km (Rural)
Data rate	DL 50 kbps UL 50 kbps	DL 50 kbps UL 980 bps	DL 600 bps UL 100 bps
Infrastructure deployment	No	Yes	No
Expansion	Low	High	High
Compatible Arduino microcontroller	MKR NB 1500	MKR FOX 1200	MKR WAN 1310

of each sensor. Finally, maintenance staff are basic users. They can view the sensor data and query flow and pressure values of damaged pipes, as well as calculate maximum repair times automatically from the information recorded in the system.

2. wAIter function modules

The proposed platform facilitates the comprehensive management of failures in WTNs, enabling the analysis of each incident from different points of view and estimates of the time available to carry out repairs. Its functional modules are described next.

2.1. Failure detection module

This module analyzes sensor pressure data to detect failures in real time in branched WTNs with a single water supply source. Likewise, no pumps may be in operation in the sector(s) where the pressure sensors are installed. Moreover, when data logging is started, the WTN must be operating correctly. The type of pressure deviation incidents and the failure detection algorithm are described next.

2.1.1. Type of incidents

There are four typical pressure variation incidents in WTNs as described below.

a) Incident 1: Sensor failure. This incident occurs when some data sending cycles are not performed correctly for any reason. Pressure data transmission failures cause erroneous pressure fluctuations that must be analyzed. Although situations of this type are rare in robust data logging devices, it is important to be aware that they may occur. Even if the duration of a fault is short, it must be identified as a sensor failure to avoid generating a false failure alert. This type of incident can be easily detected because the pressure fluctuation is associated with a fluctuation of the device battery (Fig. 3). This incident does not cause water losses.
b) Incident 2: Supply cut-off. A cut-off in water supply in the sector where the sensor is located causes the water to drop (Fig. 4). These interruptions of supply are necessary to perform cleaning and maintenance operations in the network. When the shut-off valve located upstream from the sensor closes, the pressure line shows a sudden drop in pressure from normal values to zero pressure. After maintenance is completed, the valve reopens, and the pressure returns to its normal range of values. As in the previous case, this incident does not involve water losses.
c) Incident 3: Leaks/breaks downstream from the sensor. This incident is characterized by the evolution of pressure over time when a leak occurs downstream from the pressure measuring device. Fig. 5 shows four distinct periods in the evolution of pressure data. In the first stage, the pressure is within the normal operating range and ends with a sudden drop in pressure when the leak/break occurs. In the next period, the pressure stabilizes at a lower pressure value than normal until the repair begins. The duration of this phase determines the amount of water losses caused by the leak/break in the pipe. Once the problem is detected, the upstream shut-off valve is closed and the repair period begins. The repair period ends when the pipeline is put back into service. The locations of the shut-off valve and the sensor, as well as the topology of the damaged pipe, determine the minimum pressure recorded by the sensor during the repair period. As shown in Fig. 5, the pressure returns to normal values once the fault has been fixed and the shut-off valve has been reopened.
d) Incident 4: Occurrence of leaks/breaks upstream from the sensor. This type of incident is detected in a similar manner to the previous one (Fig. 6). In this case, leak/break is detected by a sudden drop in pressure values, which continue to decrease progressively as the flow rate in the damaged pipe is reduced due to water losses. The repair period begins with the shutdown of the supply by closing the corresponding shut-off valve. The locations of the shut-off valve and the sensor, as well as the topology of the damaged pipe, determine the minimum pressure recorded by the sensor. As in the above type of incident, the pressure record values return to normal levels after reopening the shut-off valve once the leak/break has been fixed.

2.1.2. Rule-based decision algorithm for failure detection

A rule-based decision algorithm has been developed to detect failures in WTNs using pressure data analysis. This type of algorithm has a similar structure to that of a decision tree. Its behavior is based on making decisions related to the values of the input data. This methodology is a supervised self-learning algorithm for solving classification problems (Safavian & Landgrebe, 1991). Fault detection is a binary classification problem whose solution determines whether the fault exists or not.

The proposed algorithm has tree-like flowcharts in which an internal node represents a feature (or attribute), the branch represents a decision rule, and each child node represents the decision (Breiman, 2001). The nodes in the tree act as a test case for some attribute, and each branch descending from those nodes corresponds to one of the possible responses to the test case. Finally, the terminal nodes of the tree indicate the final classification. Logical structures of this type convert complex decisions into a set of several simpler decisions.

The flowchart shown in Fig. 7 is designed to determine the type of failure that occurs in a WTN due to deviations of pressure values from normal values. In the figure, the orange elements represent the nodes, the green ones represent the branches, and the blue ones represent the terminal nodes. In this case, as four possible incidents have been considered, the blue nodes identify the type of failure in the pressure records described in the previous section. The pressure and the battery of the measuring device are the attributes considered to define the branches starting from each node.

This algorithm is executed each time the database receives new data (pi(t), pressure at time t). After the algorithm is executed, the data are

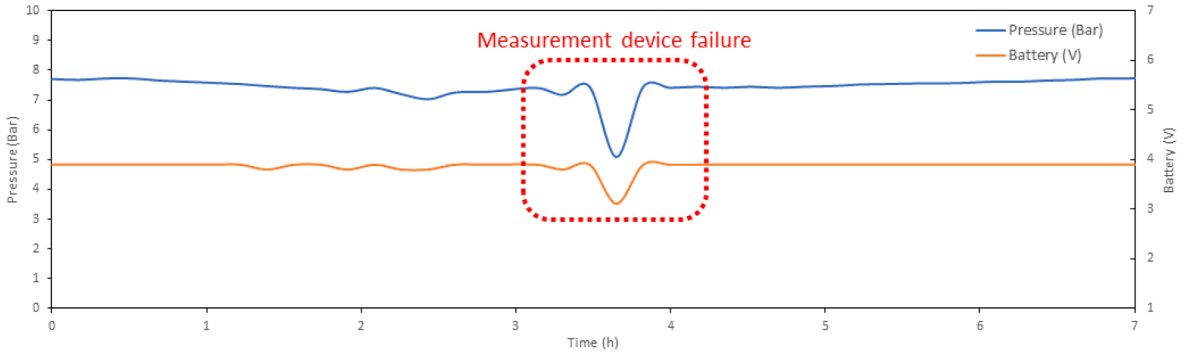

Fig. 3. Sensor failure incident.

Fig. 4. Supply cut-off incident.

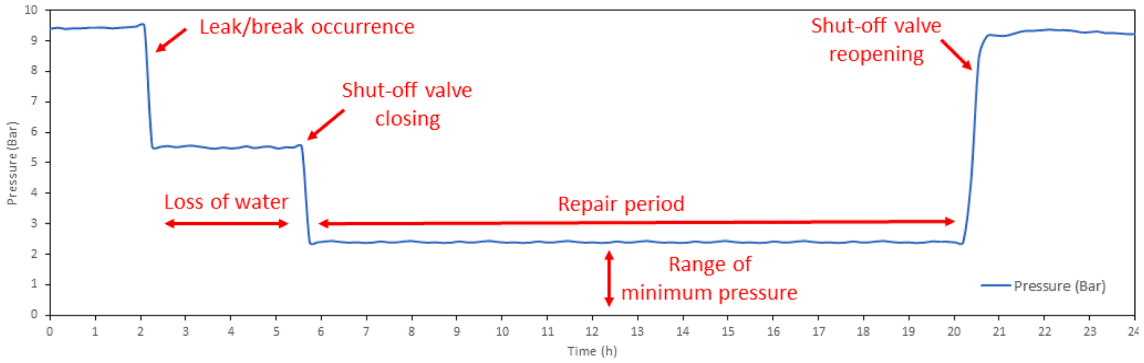

Fig. 5. Occurrence of leaks/breaks downstream of the sensor.

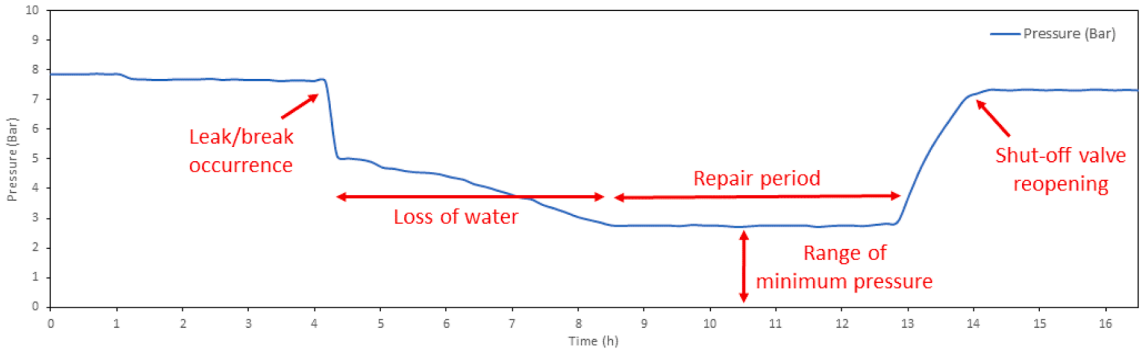

Fig. 6. Occurrence of leaks/breaks upstream of the sensor.

stored in the database. Subscript i identifies the sensor that recorded the data. Likewise, pi(t-1) is the pressure data before time t stored in the database of sensor i; pi(t + 1) is the pressure data after time t recorded by sensor i; bi(t) is the battery level of the measuring device i at a given time, t: bi(t-1) is the battery of sensor i at the instant immediately before time t.

The algorithm starts by checking whether pi(t) is below a certain pressure threshold, which is calculated in each time step (described in section 3.1.2.1). If it is under this value, it is checked whether this situation had occurred for pi(t-1). Thus, if pi(t-1) is higher than the threshold, the battery condition is checked, otherwise the failure detection process finishes. When bi(t) is not equal to bi(t-1), the sensor is not operating properly (incident 1). Then pi(t) is rejected and not stored in the database to avoid false data being recorded and no alert is sent to the WTN managers. This type of failures does not occur continuously and are of a short duration. Otherwise, if the failure occurs repeatedly, the algorithm can detect it and alerts the users to repair it.

Conversely, when bi(t) equals bi(t-1) (the sensor performance is correct) and pi(t) equals 0, the algorithm identifies that water is not circulating through the pipe because of the supply cut-off upstream of the sensor (incident 2). An alert is then generated and sent through the alert module (section 3.2) and the algorithm ends.

Finally, leaks and breaks (incidents 3 and 4) can be detected when pi (t) is not equal to 0. The algorithm sends a first notification that a leak has occurred. In the next data sending cycle, the algorithm determines whether the leak/break has occurred downstream (incident 3) or upstream (incident 4) of the sensor by comparing pi(t) to pi(t + 1). If pi(t) is almost equal to pi(t + 1), incident 3 is detected. However, if pi(t) is lower than pi(t + 1), then the leak/break has occurred upstream of the sensor location. Once the failure has been classified, a second notification is sent to the WTN managers informing them if the leak/break is located upstream or downstream of the sensor.

In addition, the algorithm detects that the pressure has returned to its ordinary values after the failure has been fixed when pi(t-1) is below the threshold, but pi(t) is above it. At that time, an alert is sent indicating that the pressure values have returned to their normal levels. In this way, the WTN managers are informed in real time when the network is back to its normal operation.

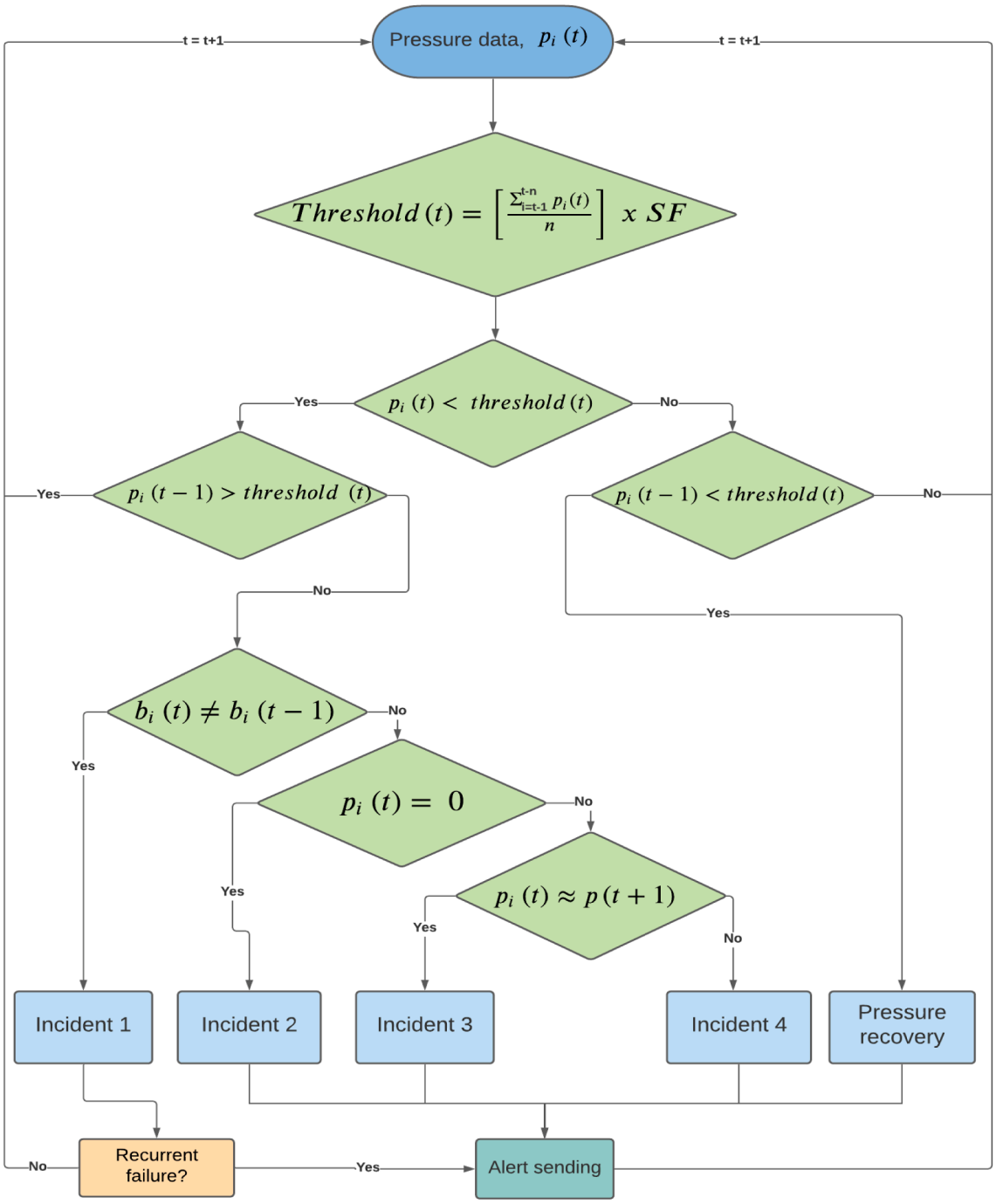

Fig. 7. Flowchart of the leak detection algorithm.

The algorithm described can be applied with a single sensor, but the denser the pressure sensor network, the easier it is to locate WTN failure points.

2.1.2.1. Threshold definition. The pressure threshold is the key parameter (attribute) for the performance of the rule-based decision algorithm described above. This parameter defines the lowest pressure limit for detecting failures in WTN when the recorded pressure is below this limit. The accuracy of the incident classifier depends on the precision of the pressure threshold determination, as it is the decision-making attribute of the first node.

The pressure changes over the year and time of day (Fig. 8). Therefore, the pressure threshold is a dynamic parameter. The threshold value is calculated as the mean of the last n pressure values recorded in the database weighted with a safety factor, SF, ranging from 0 to 1, according to Eq. (1). The shape of the threshold time evolution curve is similar to the pressure curve, since the threshold value depends on the last received pressure values, thus adapting to changes in the WTN loading conditions.

$$threshold(t) = \left[\frac{\sum_{i=t-1}^{t-n} p_i(t)}{n}\right] xSF \tag{1}$$

The values of n and SF are estimated based on the pressure variability at the monitoring points. Each sensor in the network is defined by a different SF and n. Hence, the threshold is adapted to the pressure conditions at each specific measurement point. The value of n depends

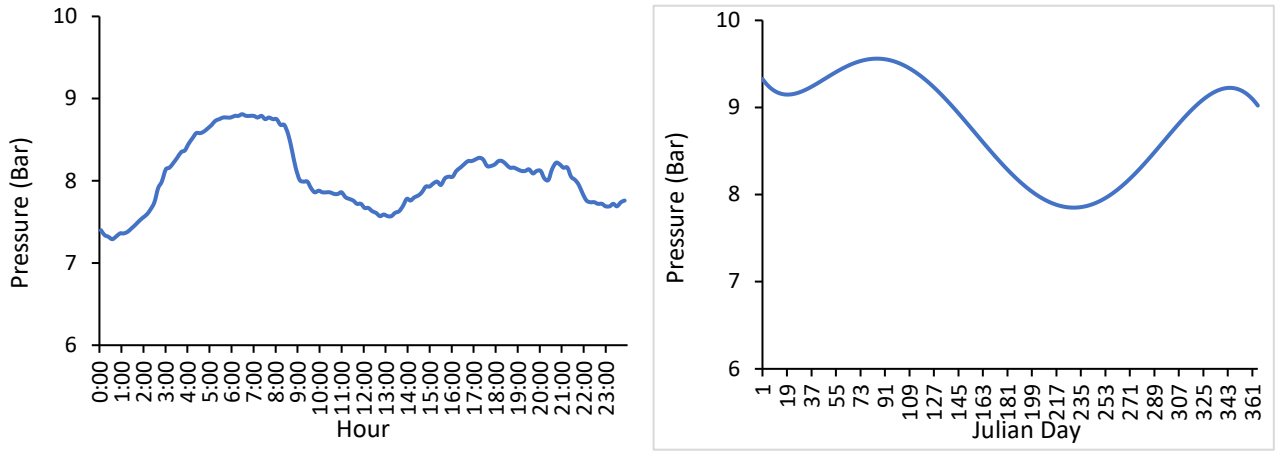

Fig. 8. Real pressure data from the ad-hoc pressure measuring device.

on the type of network to be studied. In a network with sudden pressure changes, it is necessary to set a relatively low value of n, to quickly adapt the threshold to the changes. In networks with constant pressures, a higher value of n can be set to avoid false positives of network failures. The choice of the optimal SF is a complex process that is mainly based on the performance of the algorithm (measured by the confusion matrix and the calculation of the mean detection time, MDT). This parameter is key for adapting the fault detection algorithm to the network.

The accuracy of the algorithm that calculates the SF and n parameter is evaluated by the confusion matrix as it is a classification algorithm (Fig. 9) (Fawcett, 2006; Sokolova & Lapalme, 2009). The rows of the matrix indicate the actual class and the columns indicate the predicted class. The elements that form the confusion matrix are: true positives, TP, which are the number of times the algorithm correctly detects incidents; false positives, FP, which are the number of times the algorithm detects non-existent incidents; true negatives, TN, which are the number of times the algorithm does not detect incidents because they have not occurred; and false negatives, FN, which refers to the number of times the algorithm does not detect real incidents.

The last aspect to evaluate is the mean detection time (MDT) of the incidents (Eq. (2)). MDT is the period from the beginning of any incident until it is detected by the algorithm. This parameter indicates how quickly the algorithm detects incidents in the network and is directly proportional to SF. An increase in SF will decrease the threshold that determines the occurrence of an incident and consequently increases the average time that the algorithm takes to detect it.

$$MDT = \frac{\sum_{i=1}^{N}\left(t_d^j - t_p\right)}{N_d} \tag{2}$$

Where t_d^j is the exact time that the measurement device j takes to detect an incident; t_p is the exact time the incident occurs; and N_d is the number of total incidents. High MDT values indicate delayed detection of faults. When these faults are leaks/breaks, significant amounts of water can be lost in an uncontrolled manner. Therefore, it is important to adjust the threshold value to minimize the number of false negatives and reduce MDT.

		Prediction	
		Incident	**No Incident**
Reality	**Incident**	True positives (TP)	False negatives (FN)
	No Incident	False positives (FP)	True negatives (TN)

Fig. 9. Confusion matrix.

To calculate the SF parameter, an iterative process is followed starting with a value close to 0 and increasing this parameter according to the results of the confusion matrix (TP, TN, FN and FP) and MDT. The user can define minimum requirements (TPreq, TNreq, FNreq, FPreq and MDTreq) that stop the algorithm when it exceeds them. Once the SF is optimized, the previous results are improved by performing a similar process, but this time by modifying the value of the parameter n. The objective is to improve the previous results of the confusion matrix (TP1, TN1, FN1 and FP1) and MDT (MDT1) with an optimal value of parameter n.

These two parameters, SF and n, are updated weekly to learn from new failures that occur during that period and thus optimize the calculation of the threshold. The optimization process of the parameters involved in the calculation of the threshold (SF and n) is explained in the flowchart in Fig. 10.

Fixed thresholds require a relatively large dataset to fix their value and do not detect small leaks linked to small drops in pressure records (Venkatasubramanian, Rengaswamy, Kavuri, & Yin, 2003). The proposed threshold is a dynamic threshold that updates its value with each new data sending cycle, thus adapting to the evolution over time of the WTN loading conditions (Fig. 11). With the dynamic threshold, the fault detection algorithm proposed in this work can detect small leaks that would not otherwise be detected if a fixed threshold were considered.

Fig. 11 shows the evolution of pressure and threshold during a pressure variation incident. For the duration of the incident, the threshold value remains constant at the value it had at the time of the incident. This prevents the threshold from taking excessively small values, which would not allow the identification of failures resulting from smaller pressure variations.

2.1.2.2. Algorithm performance evaluation. The parameters precision, recall, accuracy and F1score, which are calculated from the elements of the confusion matrix, are used to complete the evaluation of the algorithm (van Rijsbergen, 1979). Recall is the fraction of the true positive values that are predicted to be positive (Eq. (3)). Precision is the fraction of positive predictions that are positive (Eq. (4)). Accuracy is the ratio of correct predictions to all predictions (Eq. (5)). F1score measures the accuracy of the proposed classification algorithm as a function of precision and recall (Eq. (6)). F1score is used to minimize false positives and false negatives in unbalanced datasets.

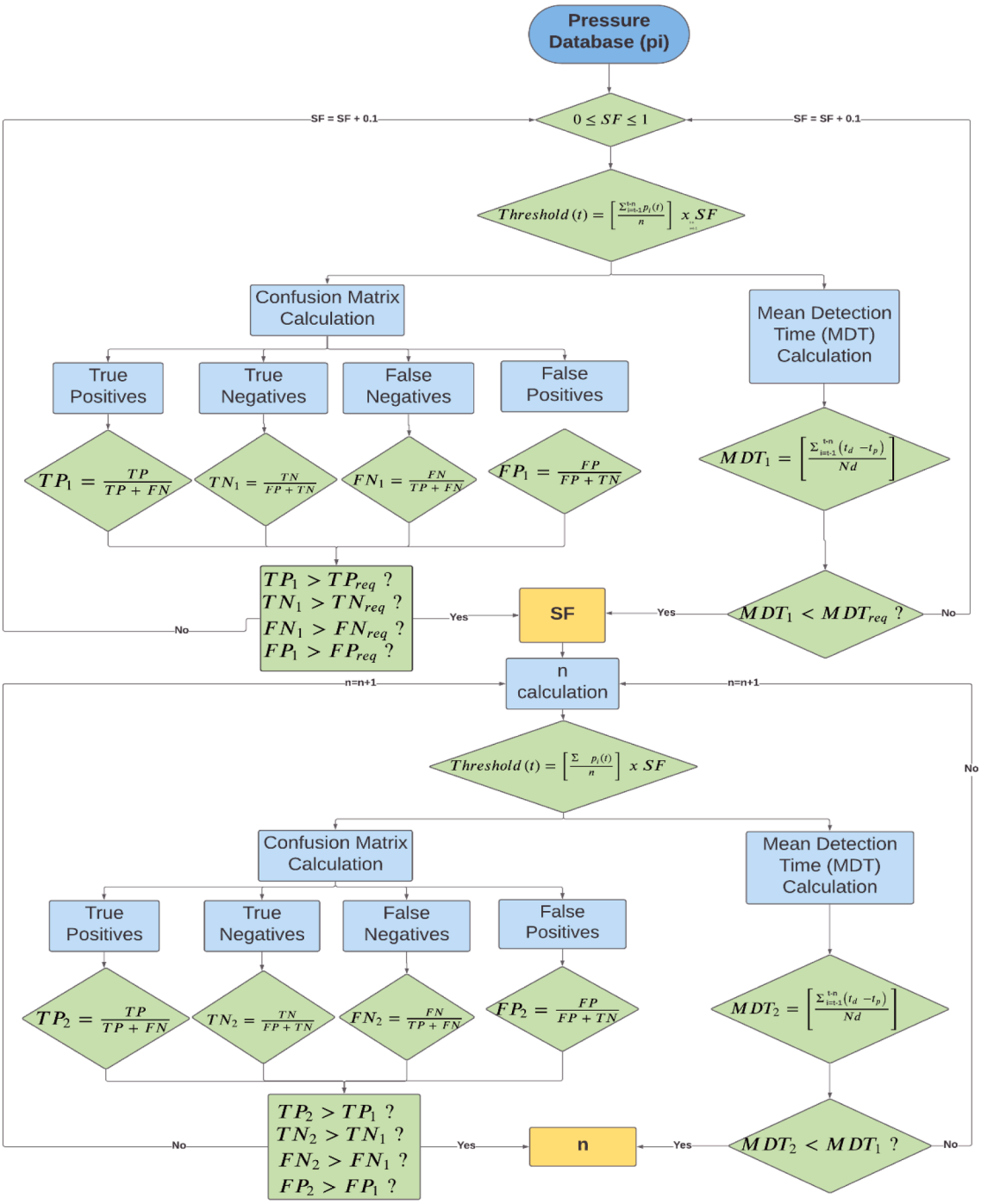

Fig. 10. Flowchart for the estimation of SF and n.

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \tag{5}$$

$$F1score = \frac{2\hat{A}\cdot precision\hat{A}\cdot recall}{precision + recall} \tag{6}$$

2.2. Alert module

This module receives the results of the fault detection module when it detects a fault and generates the corresponding alert message via SMS/email. Its purpose is to allow users to visualize the results of the failure detection module as soon as the failure occurs and to act accordingly.

This is a selective alert sending module. Alert messages are only sent to the staff responsible for the area affected by the fault. To do this, the module integrates a database with information about the maintenance personnel responsible for each sector in which a sensor has been installed. To perform this process, the sensors are georeferenced so that when the fault detection module triggers an alert, the module identifies the appropriate staff.

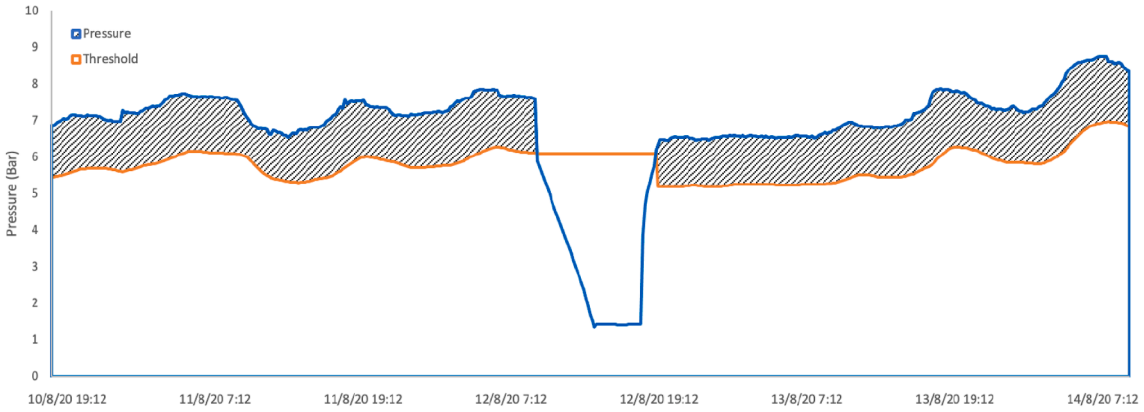

Fig. 11. Pressure and threshold evolution during a pressure variation incident.

Each alert message sends information about date and time of failure detection, recorded pressure value, type of incident and location of the sector where the failure has been detected (Fig. 12).

2.3. Failure repair management module

The purpose of this module is to determine the maximum repair time of a failure, MRT, or the duration of a maintenance task. MRT is equivalent to the interval between the time the failure is detected and the time users (inhabitants of a municipality) are unable to meet their demands because of the supply failure. This concept is the basis for the optimal management of the human and material resources needed to resolve failures. Once MTR is known, the WTN manager makes the necessary arrangements to carry out the repair, which must be solved in less time than MRT.

The consumption nodes of a WTN are the tanks that supply drinking water to the municipalities. When a supply network fails, the time to empty these tanks determines the value of MRT. To estimate MRT, it is necessary to have a hydraulic model of the WTN, which is a set of mathematical equations that simulate the behavior of the network. The hydraulic model reproduces the evolution of tank levels over time using an extended simulation period approach (Paez & Filion, 2020).

The Django Rest Framework (DRF) open-source tool is used to implement the REST API. The design of the REST API is based on three essential components: serializers, views and routers (Fig. 13). These components connect the frontend and the EPANET core. Because the frontend and the repair module are written in different programming languages, serializers are used to transform the data from EPANET into the JSON format used by DRF so that the data can be visualized using ReactJS. Views is the module in charge of the logic and manages all the communications between the API and the EPANET core. The routers define the API URLs. There is one router per view, which takes as parameters the name of the view and the URL of the API created. The computing procedure was developed in Python to use the EPANET hydraulic simulator (open-source software) as a calculation engine. A description of the hydraulic simulation process is given below.

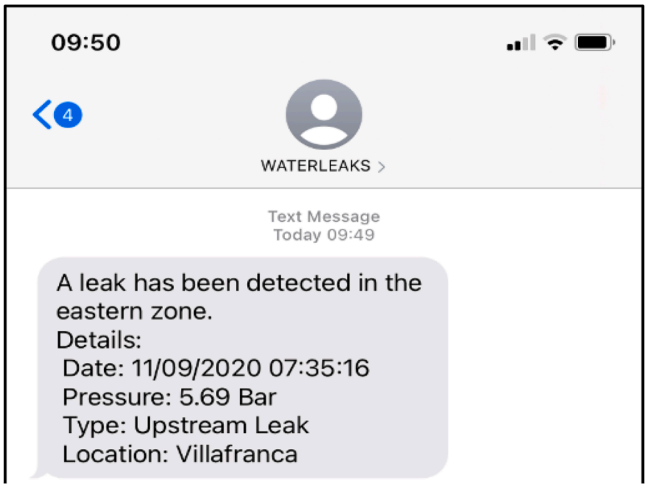

Fig. 12. Example of alert message.

2.4. MRT calculation mode

WTN managers can analyze the behavior of the network under different fault scenarios by manually entering the input data to perform the MRT calculation and simulate hypothetical fault situations in the WTN.

This calculation option is complex and requires minimal knowledge of hydraulic modelling. The only requirement to implement this option is to have the hydraulic model of the WTN. The user must manually enter the pipe affected by the failure, the level of each tank at the time of the failure and the demand of the affected municipalities. Then, using the extended period calculation of EPANET (Rossman, 2000), the maximum time available to avoid disrupting the supply of the most critical tank in the simulation scenario can be calculated.

2.5. Business solution

This section describes the design of the graphic interface of the wAIter platform. The platform includes the functionalities of the three modules explained above. A simple and intuitive interface has been designed to facilitate its replication for any water service.

In the first tab (Fig. 14), the sensors are set up to display the information they record. There is a specific button to add, edit and delete the sensors. It is necessary to enter basic information for each sensor: geographic coordinates, password to access its database, type of installation and altitude. This option is only available to the platform administrator, who is responsible for managing the sensors' basic settings. To have a background map with the layout of the georeferenced pipe network, it is necessary to upload a file with a shapefile extension containing this information.

Fig. 15 shows the location of the pressure sensors in the WTN. The platform provides users the latest recorded data of the selected measurement device by clicking on the sensor displayed on the map. With just a glance, it is possible to know the pressure in real time at the WTN monitoring sites.

Knowledge of the water pressure evolution over time in WTNs is important to gain a better understanding of how these systems operate. In many cases, historical data series are needed to analyze past time periods. The wAIter platform permits data to be consulted for a selected

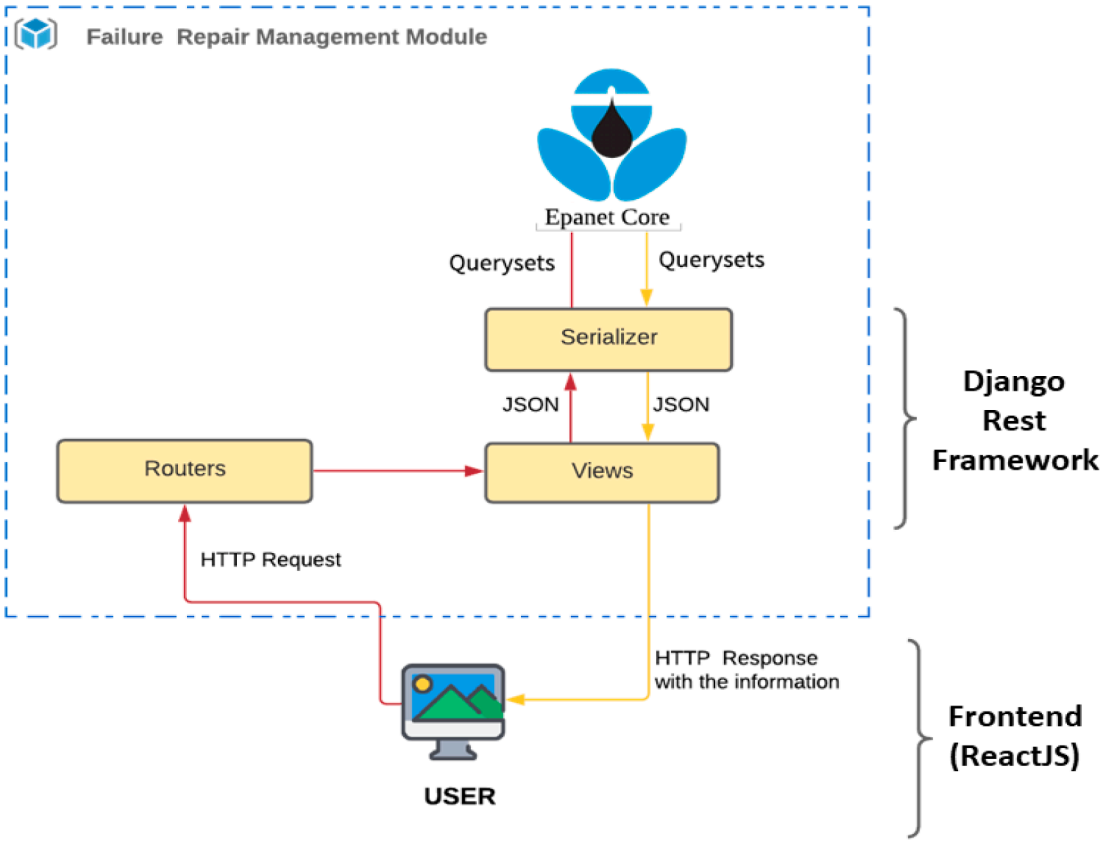

Fig. 13. Communication between the frontend and the EPANET core.

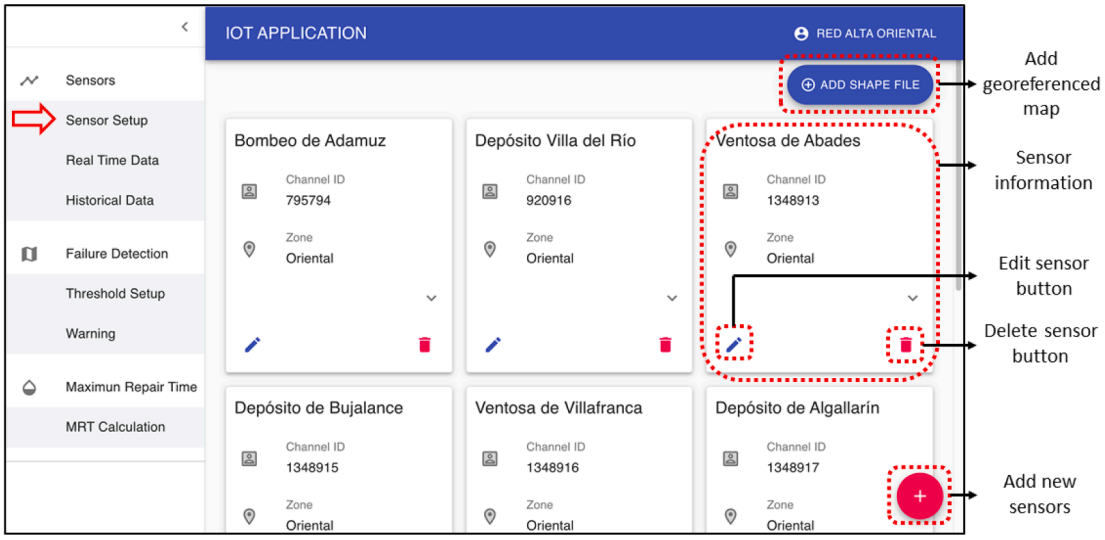

Fig. 14. Sensor configuration tab.

date interval (Fig. 16). This information can be used for the advanced calculation mode of MRT under different operational scenarios. Data can be downloaded in different formats (CSV, PDF and PNG) to adapt the query to the user's needs.

This tab configures the key parameters for the failure detection algorithm (SF and n) to calibrate the failure detection module (Fig. 17). There are two setup procedures. For sensors that have recently been installed in a location with no previous records, SF and n are entered manually. The automatic update button is for sensors that have been operating for a minimum period and have detected failures in their sector. SF and n are recalculated to take into account the last failures. Like the Sensor Setup tab, this option is only available to administrators.

The Data labelling tab has been created to label the pressure data and store detailed reports of every event occurring in the network. Data labelling is essential for fine-tuning the dynamic threshold to improve the performance of the failure detection algorithm. This way, the algorithm learns from past incidents and will self-adjust to the behavior of each network. The parameters defining the dynamic threshold (n and SF) are updated weekly according to the new data entered in the system's event database.

Fig. 15. Real time data tab.

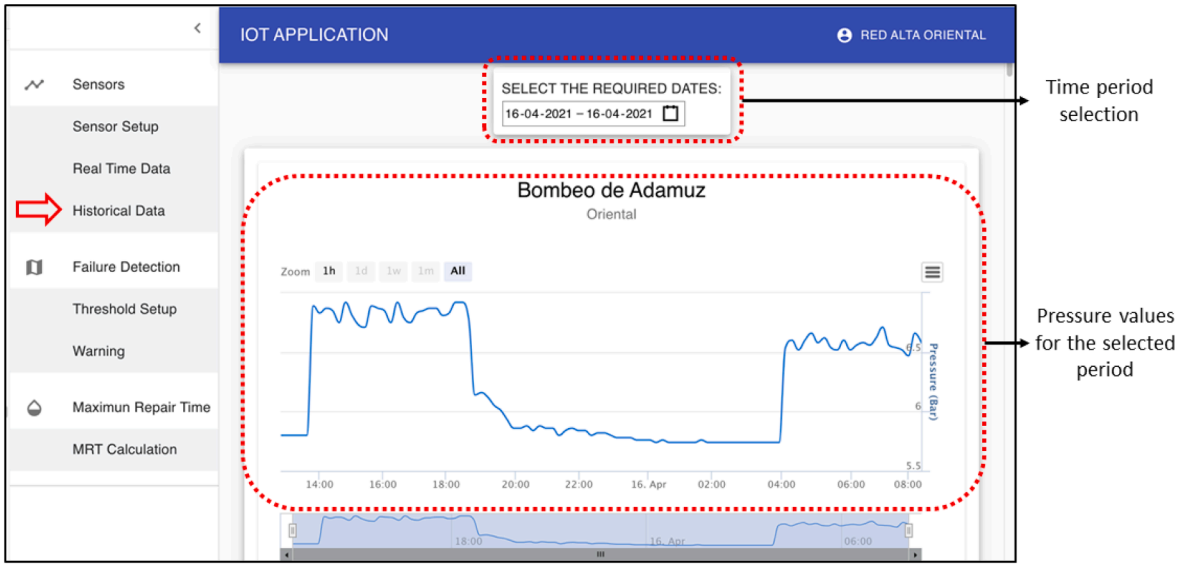

Fig. 16. Historical data tab.

To describe an incident, it is necessary to enter the starting date and time, the duration of the failure from start to full repair and the type of incident (based on the classification shown in section 2.1.1) (See Fig. 18).

The alerts tab collects information about faults detected by the system (Fig. 19). This tab is divided into two sections. The upper part of the screen contains the failure register in real time (i.e., based on the latest data collected from each sensor). The lower part of the screen contains a failure history. The platform displays the failures of the selected sensor for the selected period. This option allows users to carry out advanced studies on the recurrence of failures in the same sector.

The MRT tab is suitable for users with some knowledge of hydraulic modelling (Fig. 20). To use this function, a text file with the hydraulic model must first be created in EPANET.inp format and then uploaded by clicking on the lower right button. To create several loading conditions scenarios for each incident detected in the WTN, the hydraulic model of the network must be uploaded previously. Data on the pipeline, the level of the affected tanks and the base demand of the consumption nodes must be entered manually.

3. Results

3.1. Study area

The proposed methodology has been implemented in a WTN operated by EMPROACSA, the provincial water supply company of Cordoba, southern Spain (Fig. 21). The network covers an area of around 600 km^2 and supplies drinking water to ten municipalities with populations ranging from 140 to 9,635 inhabitants. The three largest municipalities account for 60% of the total population (44,200 inhabitants).

The Martín Gonzalo reservoir (280 masl) is the water source of this WTN. The water is purified in a water treatment plant with a capacity of 25,920 m^3/day. The population varies seasonally as many of the houses in the area are used for recreational purposes. The main industrial activity of the area is olive oil extraction, whose maximum water demand occurs during the olive harvesting period (November to February), although it only accounts for 3% of the total water consumption (Guadalquivir, 2015).

The average daily water demand is 250 l/hab/day. The mean annual

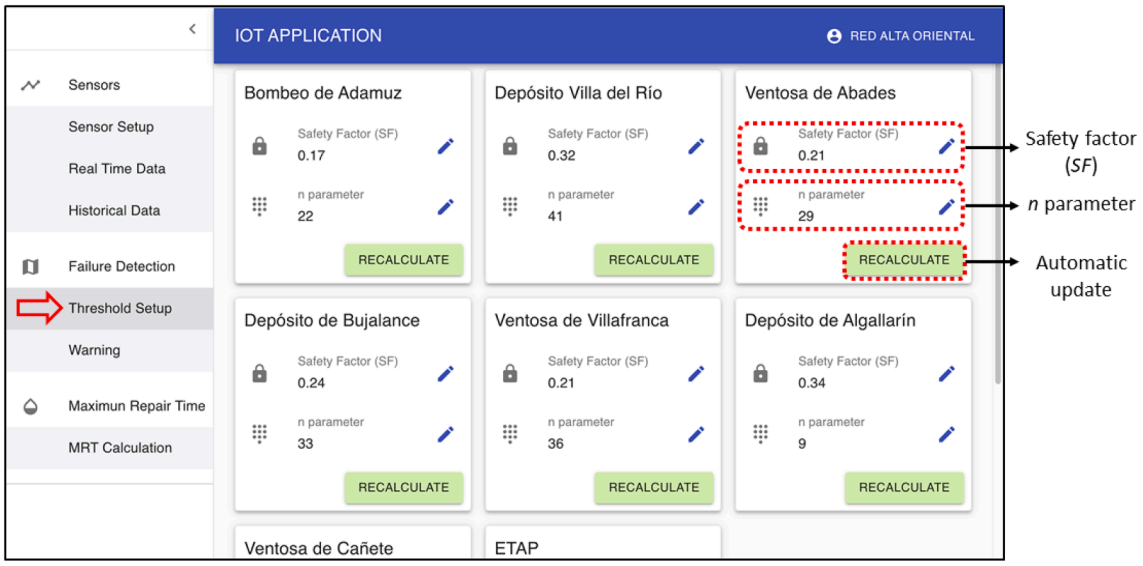

Fig. 17. Threshold setup tab (1).

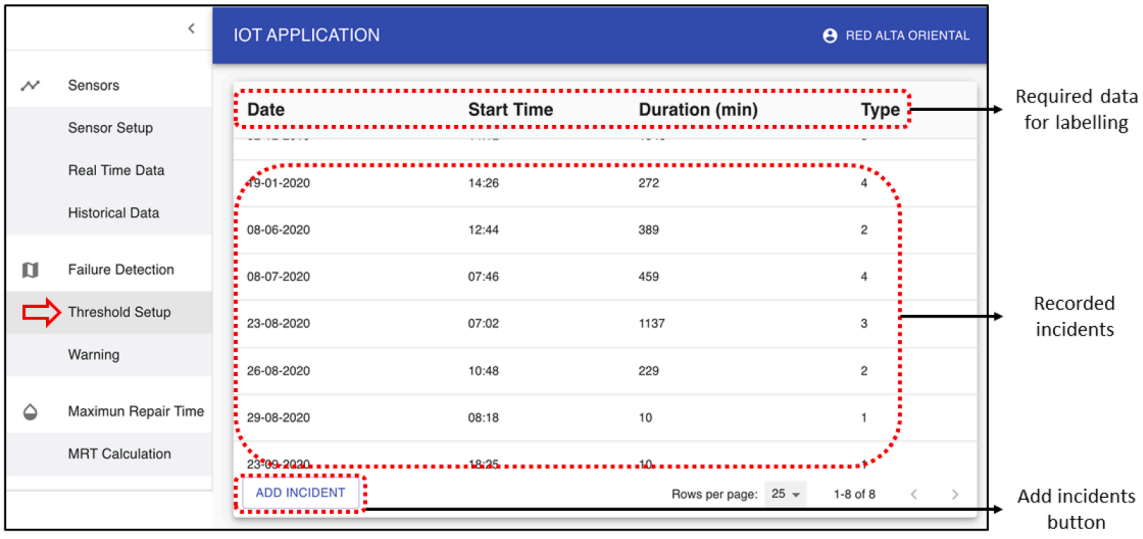

Fig. 18. Threshold setup tab (2).

consumption per municipality ranges from 12,733 m^3/year to 681,572 m^3/year and reaches the highest values in summer (July to September).

The topology of the network is branched and consists of a main pipeline with three secondary and several tertiary pipelines. The network is made up of pipes of different materials and diameters with a total length of 88.79 km. Water is pumped through 4 pumping stations with horizontal centrifugal pumps between 69 masl and 180 masl. The municipal water supply networks are fed by the 18 tanks of the WTN with capacities between 80 m^3 and 7,500 m^3.

3.2. Sensor network

Due to the state of the WTN, the Sigfox communication network has been chosen and consequently the microcontroller of the IoT sensor is Arduino MKR 1200 (Arduino, 2019). Sigfox has a range of 50 km and does not require the installation of a communications network as it can be accessed by paying the connection service with an annual fee per device. These features make Sigfox the most suitable LPWAN alternative for monitoring WTNs scattered over large territories at a low cost. A one-way communication is established between the sensor and the LPWAN network, so only the sensor readings are transmitted. This communication system is limited to a maximum of 140 messages per day and a payload of a maximum of 12 bytes. This is a sufficient frequency to monitor the water pressure and obtain data approximately every 11 min.

Eight pressure measurement devices were installed in the network. The location has been conditioned by the coverage of the Sigfox network and by the proximity to sectors with a high failure rate. The entire WTN has been covered by installing a pressure sensor in each branch of the pipe system. The technical characteristics of the pipes (diameter and material) where the sensors have been installed are varied, but have not affected the quality of the pressure records.

The core of the sensors is the pressure transducer, which measures the water pressure inside the pipes. This device converts the pressure into a voltage signal ranging from 0.33 V to 2.97 V. The supply voltage of the transducers is 3.3 V with a current consumption of 4 mA and is fully compatible with the output voltage of the microcontroller used (Arduino MKR FOX 1200). In addition, transducers have an IP69K degree of protection, which provides them protection against water and avoids damage. To improve the accuracy of the pressure measurement, two transducers with a different full scale (0–17.23 bar and 0–34.47 bar)

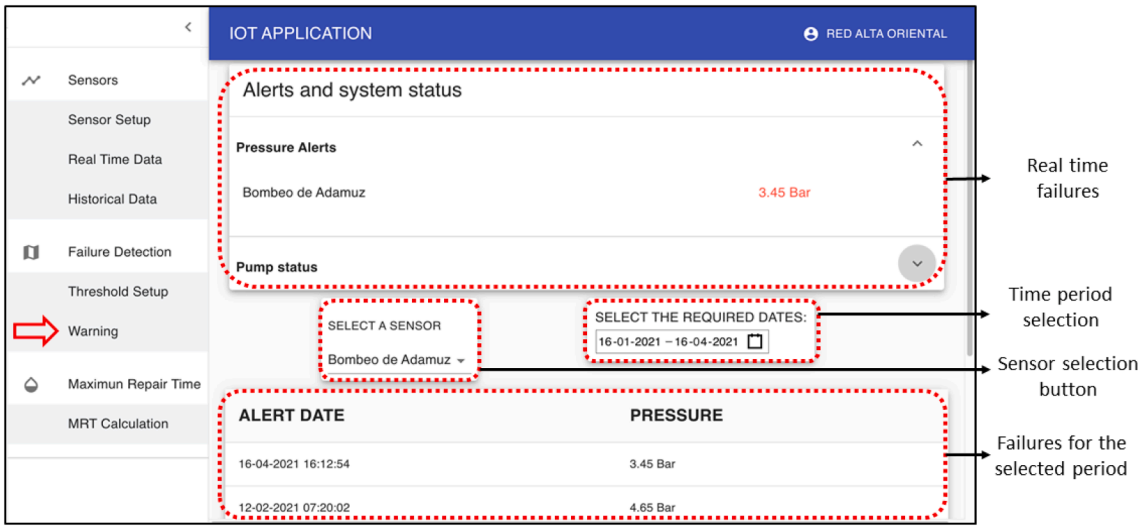

Fig. 19. Warning tab.

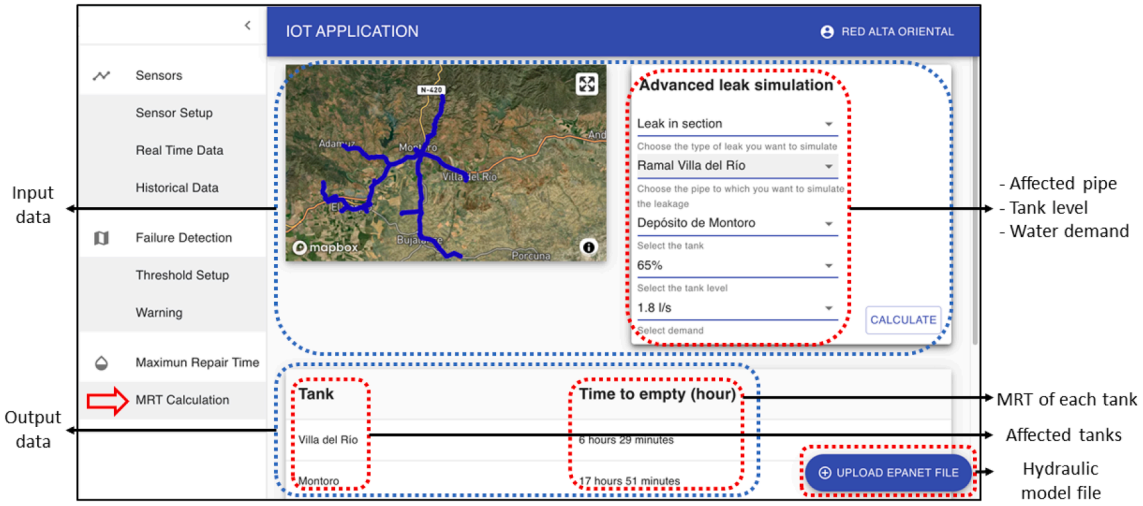

Fig. 20. Advanced MRT tab.

have been used. Thus, the pressure transducer that best suits the operating range of each monitoring point has been chosen.

3.3. *Pressure threshold calculation of the failure detection algorithm*

Using the sensor network described above, pressure data were collected from the study area over a period of 12 months. To show a more concrete case, the results will focus on one of the pressure sensors mounted on the WTN. Thus, the procedure can be extrapolated to the other sensors. In this series of pressure data recorded by this representative sensor, a total of 40 incidents (20 incidents of type 1, 5 incidents of type 2, 9 incidents of type 3 and 6 incidents of type 4) have been detected. Each of the failures is described according to its pressure, date, time and duration, location and type of incident. As an example, Fig. 22 shows how an incident can be detected by studying the evolution of the pressure. Table 2 shows how some of the incidents during the study period have been labelled to include them in the database that feeds the fault classification algorithm.

One of the key parameters in calculating the threshold of each sensor is the safety factor (SF). The data in Fig. 23 show that the optimal SF is close to 30%. This study has been carried out with data collected for 12 months, accounting for the number of false positives, false negatives and true positives with each of the SFs considered. The dashed line indicates the actual incidents that have occurred during the period analyzed. For this study, sensor failures have not been considered, as they are considered irrelevant and because these failures are due to a problem of the sensor electronics rather than the malfunctioning of the WTN.

False positives are represented by the number of times the algorithm considers an incident that has not occurred. With a low SF, there will be a high number of false positives because any small variation in pressure will be considered an incident. Conversely, a high SF implies a low rate of false positives. True positives and false positives behave symmetrically, i.e., as one increases, the other decreases proportionally. As the SF increases, the true positives decrease. This is because the threshold decreases, and the pressure drop must be greater to be detected by the algorithm. The rate of false negatives will behave in the opposite way.

The MDT, which is an important parameter for measuring the performance of the algorithm, has also been evaluated (Fig. 22). A decrease in the response times leads to a rapid action on any leak or fault and hence an improvement in the overall performance of the network.

The SF may change over time depending on the type of variable of interest to the user. For example, if the user wants to decrease the response time in summer due to water scarcity, the SF should be decreased. In contrast, false positives will increase. In addition, the

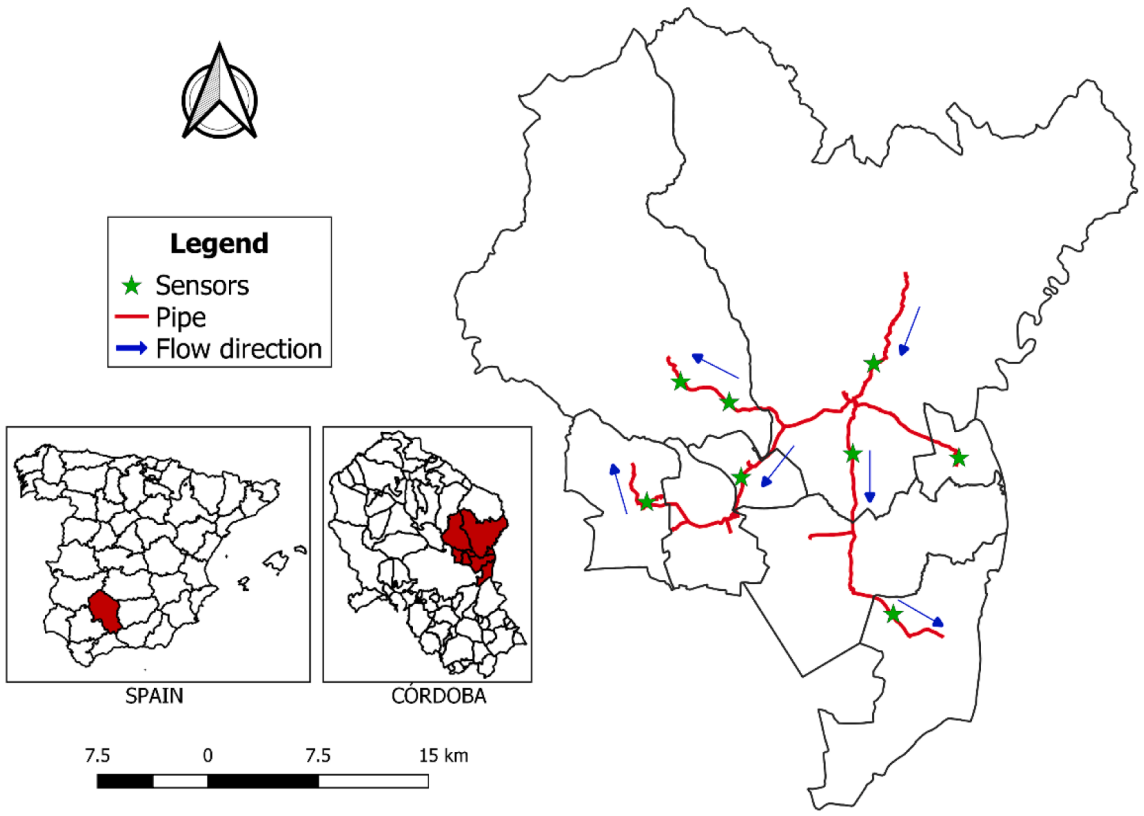

Fig. 21. Location and layout of the WTN and the sensors.

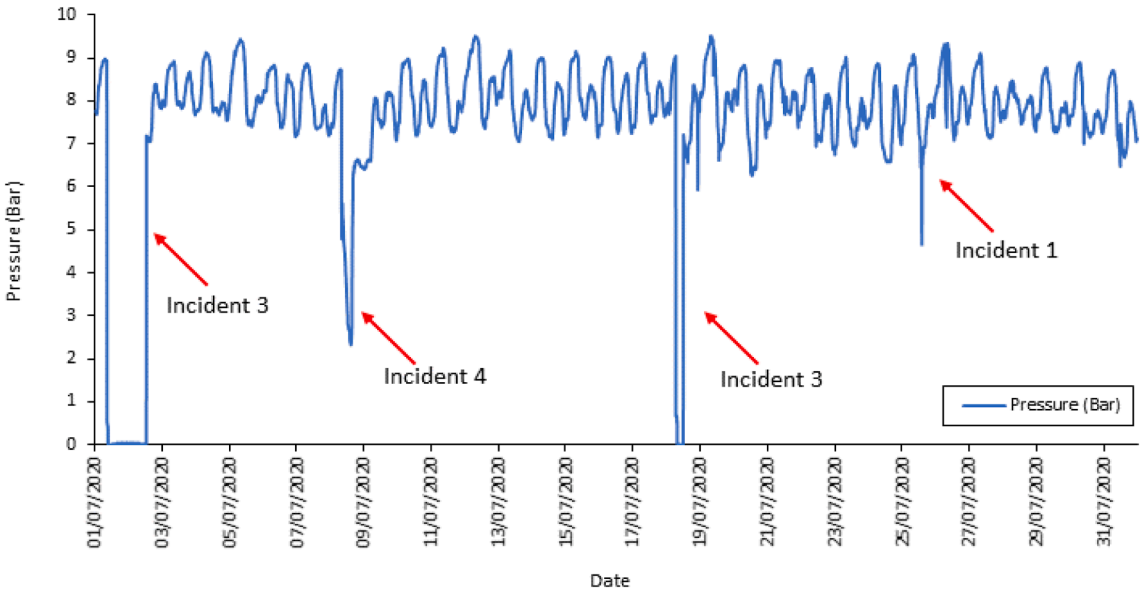

Fig. 22. Pressure evolution in July in a representative sensor.

algorithm learns from new incidents and the SF will be modified to adapt to new types of failures. An SF of 28% was chosen for this study as it detects the maximum number of incidents and provides an acceptable response time.

3.4. Performance analysis of the failure detection algorithm

Fig. 24 shows the results confusion matrix. The low rate of FN shows that few incidents were not detected by the algorithm. The high rate of TP and TN is a measure of the robustness of the algorithm and indicate that the system has not generated unnecessary alarms.

The evaluation of the algorithm performance parameters is shown in Table 3.

As can be seen in Table 3, the high value of Recall (0.95) indicates that the number of false negatives is very low. A Precision of 0.974 indicates that the number of predictions that are false positives is very low. The Accuracy value (0.999) indicates that the algorithm's predictions are almost 100% correct. The F1score is also considered since the classes

Table 2
Example of incident characterization.

Failure label	Type	Start Date (day/month/year)	Start Time (h:min)	Duration (min)	Location		
					ID	Diameter (mm)	Pipe Material
1	Incident 3	02/12/2019	14:12	1348	p512	150	Asbestos cement
2	Incident 3	23/08/2020	7:02	1137	p822	150	Cast iron
3	Incident 4	19/01/2020	14:26	272	p1019	400	Cast iron
4	Incident 4	08/07/2020	7:46	459	p77	350	Asbestos cement
5	Incident 2	08/06/2020	12:44	389	p263	150	Asbestos cement
6	Incident 2	26/08/2020	10:48	229	p471	350	Cast iron
7	Incident 1	29/08/2020	8:18	10	p512	150	Asbestos cement
8	Incident 1	23/09/2020	18:25	10	p368	150	Asbestos cement

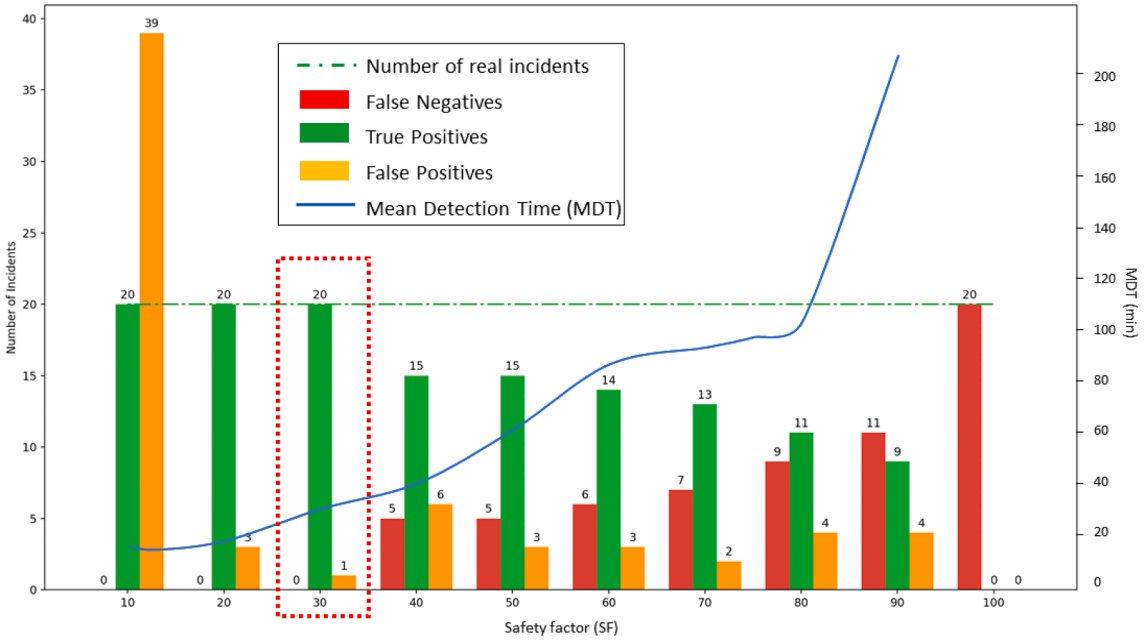

Fig. 23. Threshold safety factor and the mean detection time estimation.

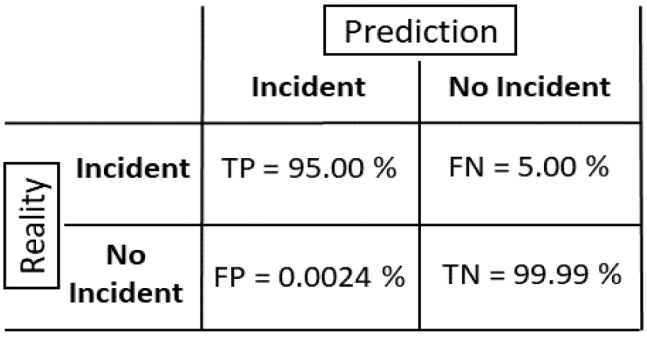

Fig. 24. Dataset confusion matrix.

Table 3
Performance parameters of the failure detection algorithm.

	Recall	Precision	Accuracy	F1score
Failure Detection Algorithm	0.950	0.974	0.999	0.962

are not balanced (only 40 events in 12 months of data). The F1score (0.962) is close to 1, showing that the algorithm is working properly.

3.5. Business solution cost for the case study

The cost of the platform and the associated sensor network has been determined considering the sensor cost and platform maintenance cost. The research and development costs of the platform and the sensor network installation have not been considered.

Each IoT pressure sensor was developed using low-cost technologies and open-source software. The total cost of each sensor was €125 for the communication node and €83 for the pressure transducer.

The number of pressure monitoring devices depends on the topology of the WTN, as well as the financial resources of the corresponding water service. The system described in this work allows for the progressive incorporation of sensors if the monitoring network is implemented in

Table 4
Annual system maintenance costs.

Concept	Price (€/year)
Sigfox (8 un.)	128.80
Database	5.60
Frontend	9.30
Servers	84.00
SMS Alert system	11.75
Total cost	239.45

phases.

The maintenance costs of the monitoring network and the platform are shown in Table 4. In this case, the sensor communication costs correspond to the system used, Sigfox. The annual connection of each device costs a maximum of €16.10, which can be reduced depending on the number of devices to be connected. On the other hand, the maintenance costs of the web infrastructure include the databases, the frontend, the servers where the hydraulic model is stored and the service for sending SMS alerts.

The total acquisition cost of the 8-sensor network and wAiter installed in the studied WTN was €1664 and the annual maintenance cost was €239.45. This amount is a low percentage of the annual maintenance costs of the WTN. Compared to other commercial alternatives, the cost of the system is 25–35% lower than the average price. In this case study, the payback period of the wAIter system is one year after its implementation due to the reduction in the network failure time. This has been calculated considering only a reduction in the volume of water lost in breaks.

4. Conclusions

The web platform presented in this work is a comprehensive support tool to manage failures occurring in WTNs. The most relevant function of this platform is the detection and classification of incidents in the network. The main elements of the proposed tool use open-source software, so water companies can control all the stages of the failure detection process (data collection, storage, analysis, visualization and warning) without the support of an external service. The platform has been successfully applied to the studied WTN during a one-year trial period.

The monitoring system linked to the web platform is fully scalable and allows increasing/decreasing the number of data collection nodes that gather information on water pressure according to the size of the WTN. Each node operates independently, so that the failure of one node does not affect the correct operation of the rest. The communication node is designed to host different communication technologies depending on the microcontroller used. The microcontroller Arduino MKR family has different boards with different communication systems, which ensures that the system will not become obsolete in the short-medium term. The platform allows accessing data from any point with an internet connection and shows the results in an intuitive and user-friendly way.

The core of the platform failure detection function is a rule-based decision algorithm. The detailed characterization of typical incidents in WTNs has allowed the development of a simple rule-based decision algorithm that efficiently detects failures in this type of hydraulic networks, as corroborated by the results obtained. The algorithm has been successfully used to classify pressure data and detect failures in the case study WTN. The performance of the algorithm depends on a dynamic pressure threshold to classify pressure records. Each pressure measurement point is defined by a different threshold. In this way, the system ensures optimal adaptation to detect pressure variations. Furthermore, the threshold is considered dynamic as the parameters defining the threshold (SF and n) can be updated to improve the detection of faults based on the faults occurring up to that moment. The proposed dynamic threshold has proven to be more reliable than traditional fixed thresholds. For the case study, the high accuracy of the fault detection algorithm (recall = 0.950, precision = 0.974, accuracy = 0.999, f1score = 0.962) was obtained with SF and n values of 0.28 and 17, respectively. Both parameters are necessary to calculate the dynamic threshold.

The MRT calculation is a useful tool for planning leakage repairs. This platform function allows optimal planning of the resources needed to carry out the repair. In this way, technical staff can simulate actual and hypothetical demand scenarios to evaluate the evolution of the tank level. When the MRT is known, it is possible to establish the maximum period for repairing faults to manage the necessary resources and prevent users from being left without supply. The proposed algorithm can be applied to WTNs with similar characteristics to the studied network, although its adaptation to other types of networks (looped networks, pumping, distribution networks, etc.) needs to be studied.

When working with real datasets of water transmission network pressure, it is time consuming to collect long series of incidents. So far, good results have been obtained with the existing data, although the system will be improved with the addition of new failure records. wAiter has great potential for future developments such as incorporating flow and level sensors in the tanks and the addition of a decision-making module based on the analysis of historical failure records to determine pipe renewal criteria according to the frequency and magnitude of failures.

CRediT authorship contribution statement

José Pérez-Padillo: Methodology, Data curation, Writing – original draft, Writing – review & editing. **Francisco Puig:** Software, Writing – original draft, Formal analysis. **Jorge García Morillo:** Validation, Supervision, Resources. **Pilar Montesinos:** Validation, Project administration, Resources.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been carried out in the framework of a collaboration agreement between the Provincial Water Company of Cordoba (EMPROACSA) and the Hydraulics and Irrigation group of the University of Cordoba. The authors are grateful to Aguas de Córdoba for the material and human resources provided to carry out this project. Funding for open access charge: Universidad de Córdoba / CBUA.

References

Al-Khomairi, A. (2008). Leak detection in long pipelines using the least squares method. *Journal of Hydraulic Research, 46*(3), 392–401. https://doi.org/10.3826/jhr.2008.3191

Apostol, E.-S., Truică, C.-O., Pop, F., & Esposito, C. (2021). Change Point Enhanced Anomaly Detection for IoT Time Series Data. *Water, 13*(12), 1633. https://doi.org/10.3390/w13121633

Arduino. (2019). *Arduino MKRFOX 1200*. https://www.arduino.cc/en/Main.ArduinoBoardMKRFox1200.

Breiman, L. (2001). Random Forests. *Machine Learning, 45*, 5–32. https://doi.org/10.1023/A:1010933404324

Butler, D., Farmani, R., Fu, G., Ward, S., Diao, K., & Astaraie-Imani, M. (2014). A new approach to urban water management: Safe and sure. *Procedia Engineering, 89*, 347–354. https://doi.org/10.1016/j.proeng.2014.11.198

Cesana, M., & Redondi, A. E. C. (2017). IoT Communication Technologies for Smart Cities. In *Designing, Developing, and Facilitating Smart Cities* (pp. 139–162). https://doi.org/10.1007/978-3-319-44924-1

Chen, M. I. N., Member, S., Miao, Y., Hao, Y., Member, S., Hwang, K. A. I., & Fellow, L. (2017). Narrow Band Internet of Things. *IEEE Access, 2*, 20557–20577. https://doi.org/10.1109/ACCESS.2017.2751586

Cody, R. A., Tolson, B. A., & Orchard, J. (2020). Detecting Leaks in Water Distribution Pipes Using a Deep Autoencoder and Hydroacoustic Spectrograms. *Journal of Computing in Civil Engineering, 34*(2), 04020001. https://doi.org/10.1061/(asce)cp.1943-5487.0000881

Django. Django: the web framework for perfectionists withdeadlines. (2020). https://www.djangoproject.com.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters, 27*(8), 861–874. https://doi.org/10.1016/j.patrec.2005.10.010

Fereidooni, Z., Tahayori, H., & Bahadori-Jahromi, A. (2020). A hybrid model-based method for leak detection in large scale water distribution networks. *Journal of Ambient Intelligence and Humanized Computing, 0123456789*. https://doi.org/10.1007/s12652-020-02233-2

Fuchs, H. V., & Riehle, R. (1991). Ten years of experience with leak detection by acoustic signal analysis. *Applied Acoustics, 33*(1), 1–19. https://doi.org/10.1016/0003-682X(91)90062-J

Gackenheimer, C. (2015). Chapter1: What Is React?. In *Introduction to React* (pp. 1–20). https://doi.org/10.1007/978-1-4842-1245-5_1

Guadalquivir, C. hidrográfica del. (2015). *Propuesta del Plan Hidrológico de la Demarcación Hidrográfica del Guadalquivir*.

Hernandez, P., & Kenny, P. (2010). From net energy to zero energy buildings: Defining life cycle zero energy buildings (LC-ZEB). *Energy and Buildings, 42*(6), 815–821. https://doi.org/10.1016/j.enbuild.2009.12.001

Herrera, M., Ferreira, A. A., Coley, D. A., & De Aquino, R. R. B. (2016). SAX-quantile based multiresolution approach for finding heatwave events in summer temperature time series. *AI Communications, 29*(6), 725–732. https://doi.org/10.3233/AIC-160716

Kumar Polu, S. (2018). OAuth based Secured authentication mechanism for IoT Applications. *International Journal of Engineering Development and Research, 6*(4), 2321–9939. https://doi.org/10.1109/ICTC.2015.7354740

Lalle, Y., Fourati, L. C., Fourati, M., & Barraca, J. P. (2019). A Comparative Study of LoRaWAN, SigFox, and NB-IoT for Smart Water Grid. *2019 Global Information Infrastructure and Networking Symposium. GIIS, 2019*, 1–6. https://doi.org/10.1109/GIIS48668.2019.9044961

Makropoulos, C., & Savic, D. A. (2019). Urban Hydroinformatics: Past, Present and Future. *Water, 11*(1959). https://doi.org/10.3390/w11101959

McKenzie, R., & Seago, C. (2005). Assessment of real losses in potable water distribution systems: Some recent developments. *Water Science and Technology: Water Supply, 5*(1), 33–40. https://doi.org/10.2166/ws.2005.0005

Narayanan, L. K., Sankaranarayanan, S., Rodrigues, J. J. P. C., & Lorenz, P. (2020). Multi-agent-based modeling for underground pipe health and water quality monitoring for supplying quality water. *International Journal of Intelligent Information Technologies, 16*(3), 52–79. https://doi.org/10.4018/IJIIT.2020070103

Paez, D., & Filion, Y. (2020). Water Distribution Systems Reliability under Extended-Period Simulations. *Journal of Water Resources Planning and Management, 146*(8), 04020062. https://doi.org/10.1061/(asce)wr.1943-5452.0001257

Pérez-Padillo, J., Morillo, J. G., Ramirez-Faz, J., Roldán, M. T., & Montesinos, P. (2020). Design and implementation of a pressure monitoring system based on iot for water supply networks. *Sensors (Switzerland), 20*(15), 1–19. https://doi.org/10.3390/s20154247

Purnama, A. A. F., & Nashiruddin, M. I. (2020). Sigfox-based internet of things network planning for advanced metering infrastructure services in urban scenario. *Proceedings - 2020 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology, IAICT 2020*, 15–20. https://doi.org/10.1109/IAICT50021.2020.9172022.

Python Software Foundation. Python Language Reference. Version 3.6.11 Available at http://www.python.org. (n.d.).

Rijsbergen, C. J. van. (1979). *Information retrieval*. https://doi.org/doi.org/10.1002/asi.4630300621.

Romano, M., Kapelan, Z., & Savić, D. A. (2014). Automated Detection of Pipe Bursts and Other Events in Water Distribution Systems. *Journal of Water Resources Planning and Management, 140*(4), 457–467. https://doi.org/10.1061/(asce)wr.1943-5452.0000339

Rossman, L. A. (2000). EPANET 2 Users Manual. *National Risk Management Research Laboratory, USEPA*, *September*, 0–200. https://doi.org/10.1177/0306312708089715.

Safavian, S. R., & Landgrebe, D. (1991). A Survey of Decision Tree Classifier Methodology. *IEEE Transactions on Systems, Man and Cybernetics, 21*(3), 660–674. https://doi.org/10.1109/21.97458

Semtech Corporation. Lora Overview. (n.d.). Retrieved September 12, 2019, from https://www.semtech.com/lora.

Shao, Y., Li, X., Zhang, T., Chu, S., & Liu, X. (2019). Time-series-based leakage detection using multiple pressure sensors in water distribution systems. *Sensors (Switzerland), 19*(14). https://doi.org/10.3390/s19143070

Singh, R. K., Puluckul, P. P., Berkvens, R., & Weyn, M. (2020). Energy consumption analysis of LPWAN technologies and lifetime estimation for IoT application. *Sensors (Switzerland), 20*(17), 1–22. https://doi.org/10.3390/s20174794

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing and Management, 45*(4), 427–437. https://doi.org/10.1016/j.ipm.2009.03.002

Soldevila, A., Blesa, J., Tornil-Sin, S., Fernandez-Canti, R. M., & Puig, V. (2018). Sensor placement for classifier-based leak localization in water distribution networks using hybrid feature selection. *Computers and Chemical Engineering, 108*, 152–162. https://doi.org/10.1016/j.compchemeng.2017.09.002

Venkatasubramanian, V., Rengaswamy, R., Kavuri, S. N., & Yin, K. (2003). A review of process fault detection and diagnosis. *Computers & Chemical Engineering, 27*(3), 327–346. https://doi.org/10.1016/s0098-1354(02)00162-x

Winkler, D., Haltmeier, M., Kleidorfer, M., Rauch, W., & Tscheikner-Gratl, F. (2018). Pipe failure modelling for water distribution networks using boosted decision trees. *Structure and Infrastructure Engineering, 14*(10), 1402–1411. https://doi.org/10.1080/15732479.2018.1443145

Wu, Z. Y., Sage, P., & Turtle, D. (2010). Pressure-Dependent Leak Detection Model and Its Application to a District Water System. *Journal of Water Resources Planning and Management, 136*(1), 116–128. https://doi.org/10.1061/(asce)0733-9496(2010)136:1(116)