SAMIoT – Middleware based on IoT for Irrigation Planning on Large-scale Crops

Jose Luis Rivas Viedman, Juliana Sánchez Benítez, Javier Alí Carbonell Gonzáles, Edgar Hincapié
Department of Information Technology and Networks
Colombian Sugarcane Research Center (CENICAÑA)
Cali, Colombia
email: {jlrivas, jsanchez, jacarbonell, ehincapie}@cenicana.org

Armando Ordóñez
Intelligent Management Systems
University Foundation of Popayan
Popayán, Colombia
email: jaordonez@unicauca.edu.co

Hugo Ordoñez
Faculty of Systems Engineering
University of San Buenaventura Cali
Cali, Colombia
email: haordonez@usbcali.edu.co

***Abstract*— Monitoring of large-scale crops may require Wireless Sensor Networks to collect information in real time. These networks require the integration of diverse devices and sensors. Middlewares have been used to facilitate the integration of heterogeneous applications and services. This article presents SAMIoT, a middleware aimed at improving the production of sugarcane through the optimization of water use. The middleware monitors the matric potential of the soil to determine the right moment to irrigate. A solution based on the proposed middleware was developed for the sugarcane crops in the Cauca River Valley in Colombia.**

Keywords-middleware; Internet of Things (IoT); Wireless Sensor Networks (WSNs); matric potential; efficient irrigation.

I. INTRODUCTION

Large-scale farming can be extended on large expanses of land. Monitoring these crops may require the deployment of Wireless Sensor Networks (WSN) based on the Internet of Things (IoT). These networks interconnect a large number of sensors to collect data from the crops in real time [1]. Obtaining an increasing amount of information from the plants enables farmers to make quick and accurate decisions.

In the sugar crops, some variables, such as the electrical resistance [2], the capacity of water absorption and the degree or thermal level of the soil are continuously monitored. These variables are used to calculate the matric potential or volumetric water content of the soil. The matric potential allows determining the most appropriate moment for irrigation. Through this process, an efficient use of the water can be achieved, with the consequent impact on the environment and the profitability of farmers.

Various methods have been proposed for the irrigation planning. Some methods are based on the quantification of components of the water balance. This quantification allows estimating a permissible level of water exhaustion in the soil. Other methods are based on monitoring the water of the soil or plants [3].

Cenicaña is a Colombian entity dedicated to the research of sugarcane. Cenicaña has tested and recommended other methods for the irrigation planning such as the use of the Cenirrometer tank [4], the installation of groundwater observation wells, and the verification of soil humidity [5].

The water balance is the most used method among cane growers in the Cauca River valley. This method requires incorporating aspects related to the hydrophysical properties of soils. However, in the Cauca River valley, there is a high heterogeneity of soils (about 238 soil varieties) and a high variability of precipitation. This variability makes it necessary to have humidity and granular matrix sensors to continuously monitor the energetic state of the water in the soil, thus improving the accuracy of irrigation planning.

These methods allow to trigger alarms according to the amount of water available for the plant [6] or with an optimum soil humidity (i.e., between ± -20 and ± -80 kilopascals (KPa)) [7].

To achieve an efficient use of water, it is necessary to determine the consumption and availability of water required for crops. IoT offers an alternative to automate and optimize data capture.

ISBN: 978-1-61208-622-4

WSNs are used in agriculture to address different problems. Most of these networks use a single system or application or are deployed only in a particular area of interest [8].

The sensors used in agriculture are not endowed with processing. The middleware presented here is called SAMIoT and allows communication between these sensors within a WSN [9]. These nodes are connected to Smart Agricultural Nodes in an IoT environment. This network can handle near real-time events, through TCP (Transmission Control Protocol) sockets supported by the modular extension of Node.js [10].

SAMIoT allows the integration of elements to facilitate the configuration of a device in the IoT ecosystem. The system enables users to manage and configure new conditions, send commands to the node, collect information from the sensors connected to this node, and send data to a server. The middleware includes notification services that inform if the values received from the sensor exceed a predefined min-max threshold value.

The middleware was applied to monitor variables that contribute to reducing the uncertainty that exists in the irrigation planning. Due to resource constraints of the project, the middleware described here is validated in a crop with six nodes; however it is designed and is envisaged to be used in large-scale crops.

The rest of this paper is organized as follows. Section 2 presents the related work. Section 3 describes the architecture of the middleware. Section 4 details the implementation of the prototype and Section 5 concludes.

II. RELATED WORK

Some tools can be found on the market for IoT development, such as ReMMoC (A Reflective Middleware to Support Mobile Client Interoperability) [11], OpenIoT (Open source blueprint for large scale self-organizing cloud environments for IoT applications) [12] and WebNMS [13]. Most of these tools focus on solutions for general use in homes and urban areas, leaving aside the restrictions or limitations of the agricultural environment. Some have focused on providing interoperability and reconfiguration to mobile applications [14]. Other works present architectures for the deployment of services in a semantic environment [15] like LinkSmart (formerly known as HYDRA). This latter approach allows the creation of ambient intelligence (AmI) applications, through the combination of service-oriented architecture (SOA) and an architecture based on semantic models.

ReM-MoC [11] is a reflective middleware platform that dynamically adapts its link and protocol to allow interoperability with heterogeneous services. Also, the ReMMoC programming model is integrated into the platform. This model is based on the concept of Web Services of abstract services for the development of mobile applications through a middleware platform.

Recently there has been a need to give users control over the IoT. This requirement increases the importance of middleware since it simplifies the development of new services and technologies or integrates them with existing ones. Hence, the importance of projects like OpenIoT [12] (FP7-287305) that develop a middleware platform allowing the configuration of utility-based applications, as well as cloud-based detection services.

Platforms such as WebNMS [13] seek to help companies achieve their IoT / M2M (Machine to Machine) objectives. WebNMS allows making timely decisions and optimizing processes. WebNMS supports customization and extensibility between domains. Despite its advantages, this tool is aimed at professionals with high knowledge in software development.

Regarding agriculture, few studies can be found that show the development of middlewares to support a large number of sensors. In [14], it is shown the digitization of some aspects of agriculture such as agricultural production, livestock, aquatic products and forestry industry through Information Technology, Internet connectivity, and the IoT. Also, in the field of agriculture, it is widespread to find applications dedicated to the monitoring of the environment in various disciplines, such as, control of livestock sheds, and precision agriculture.

On the other hand, CarrIoTs [16] presents a cloud-based platform service for IoT projects. This platform supports M2M communications and focuses on scalability at the network level. CarrIoTs does not guarantee storage security and offers limited interoperability. Table I shows the gaps between five alternatives to develop IoT solutions.

TABLE 1. RELATED WORKS (GAPS)

	Service Discovery	Cloud deployments	On-premises deployments	Unlimited heterogeneous frames /sensors	Open source
ReMMoC	✓		✓		✓
WebNMS		✓			
OpenIoT	✓	✓	✓	-	✓
Carriots	✓	✓			
Hydra / LinkSmart	✓	✓			✓

Most of them are based on services discovery, which limits the support to multiple frames or custom frames. On the other hand, some limitations do not allow implementation on various servers.

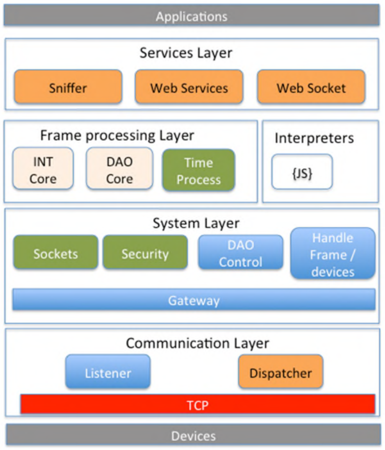

Figure 1. Middleware architecture.

III. ARCHITECTURE OF THE MIDDLEWARE

The architecture components are described below (see Fig 1).

A. *Architecture*

the proposed middleware is based on two architectural approaches for middlewares design [17]:

Message-Oriented Middlewares (MOM): here the communication is based on messages that include extra metadata. Compared to event-oriented middlewares, messages are bearers of sender and receiver addresses. In MOM, the listening engine of the proposed middleware receives and processes packets whose frame structure and sender information have been previously defined at the database level.

Middleware based on agents: Since the Internet does not have a single solution for all domains this approach aims at providing some features such as resource management, reduction of network load, code management, asynchronous and autonomous execution, availability, robustness, fault tolerance and adaptability. In this approach, applications are divided into modular programs called agents, to facilitate the injection of events and their distribution through the network, facilitating the design of decentralized systems. In this middleware, agents are represented on a logical layer based on interpreters. This representation makes it possible to offer a fully configurable and extensible middleware solution for the IoT. This solution is also adaptable to different operating environments, and frames.

B. *Design*

Fig. 2 shows a general abstraction of the architecture showing the communication dependencies between components. The four layers of the architecture are described below:

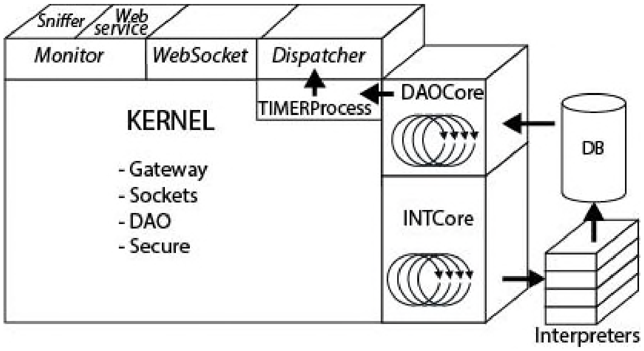

Figure 2. Abstraction of middleware components

1) Communications layer: This is responsible for establishing, controlling, maintaining communication to and from the devices. The *Listener* process has a permanent listening port, while the *Dispatcher* is responsible for sending the commands defined by the users or applications to the *Smart Agricultural Nodes*. These commands can be sent through a console or can be retrieved from commands stored in the database.

2) *System layer:* This layer is responsible for providing technical and logical interoperability between the smart agricultural node and the middleware. According to ETSI (European Telecommunications Standards Institute), technical interoperability allows linking hardware, software, systems and platforms components to establish a machine-to-machine communication. This layer focuses on the communication protocols and the necessary infrastructure so that they can interoperate. This layer also provides the objects that will be instantiated in each interpreter. Besides, this layer also carries out the frame and checksum validation, manages the DAO (Data access Object) layer and various interpreters.

At the security level, there are two primary processes. Firstly, the MOM is used to send the sender's IP address as well as the serial number or identification registered in the database in the extra data. In this way, only the connection between a node and the middleware is allowed. Secondly, once the connection is established, and the frame is received, this frame is sent to a security process that verifies if the type of incoming frame has been previously registered in the database. Once a satisfactory response is obtained, the quality of the data is validated with the value of the checksum of the frame. On the other hand, the security of the transmission is left to the TCP / IP protocol.

3) *Frame Processing layer*: This layer is made up of three processes

DAOCore: This layer is based on the DAO pattern that makes the logic of the middleware independent of the persistence, reduces the transaction complexity and improves the performance of container-managed transactions. This pattern enables interpreters to interact with different databases. The middleware for the information capture requires a minimum structure at the database level. In addition to storing data from the collected variables, this database stores several types of valid frames (table 2). Likewise, the names, IP or MAC addresses of the Smart Agricultural Nodes, devices or datalogger are stored to comply with the MOM approach. Finally, the topic identifiers for each WebSocket are also stored.

INTCore: This process manages the interpreters. These interpreters process and parse the diverse message formats (frames). These frames are received from the clients and defined in the database.

TIMERProcess: This process is permanently monitoring the rules in the database. When a condition is met, this is processed and dispatched to the smart Agricultural Node through the command dispatcher. This sending can be done to retrieve lost data due to connection failures, or to activate actuator devices (open or close the water valves, turn a water pumping on or off).

4) *Services Layer:* This layer is composed of the following elements:

WebSockets: provides a bi-directional channel of real-time communications for advanced web services and applications. Here, the information from each socket/thread is exposed as a topic and may be consumed by different clients. This method provides a near real-time communication.

Sniffer: This component exposes some events that are occurring within the kernel. Fundamentally it is a TCP service that allows a connection through a Telnet client.

WebServices: Exposes an API to query information or trigger core events. The exposed services are:

- list: list the connected devices.
- Broadcast: Send a message to all connected devices.
- closeByIp: close the connection with a device using the IP address.
- close: Close all client connections.
- closeAll: Close all client connections and restart the core server.
- ReloadInterpreters: restart the interpreter driver and set a newly created interpreter in the middleware.
- sendByIp: Send commands to the Smart Agricultural Nodes.

5) *Interpreters*

These are routines in JavaScript that process each frame format in a customized way. These methods create instances of the CORE processes, are autonomous and preventive, and translate the frames sent from the INTCore. The user of the middleware can modify two classes. The first to perform the parsing of the frame and to process the information (apply equations, correct data, etc.). The second class is used to adjust the fields to the model before inserting the information into the database.

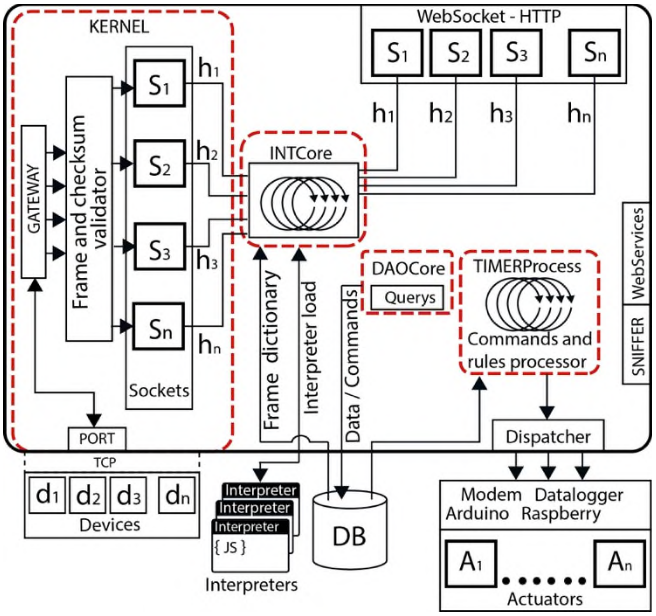

Figure 3. Operation flow of middleware.

Fig. 3 shows the communication diagram of the components. The middleware is listening permanently for a specific port. The clients or devices (dn) are connected to the server using this port via TCP/IP. A TCP socket (sn) is created per client. Each socket is handled by a thread (hn), thus, simultaneous communications with multiple devices are supported.

The information that circulates in each thread is passed to the interpreter, which parses the data according to the pre-established information for each type of device. Thus, a level of security is generated to the frames, since the frames not defined in the database are discarded. On the other hand, the probability of critical neck in the communication flow or an overload is very low because this process is managed by the event loop of Node.js, this method is based on [18]. Once the frame is validated and processed, the data is exposed through a WebSocket service thus offering near real-time communication. Simultaneously the information is sent to the database for storage.

To be accepted, the messages must be identified by a unique type of message according to the frame format defined by the user. Table 2 presents examples of messages received before being processed by interpreters where:

TABLE II. FRAME FORMATS SUPPORTED BY THE MIDDLEWARE

Id	Type	Format
101	Meteorological	101,81508,2017-02-28,14:11:00,26.83,"2017-02-28 14:10:20",26.85,27,"2017-02-28 14:10:10",0.34,52.68,"2017-02-28 14:10:50",53.13,53.56,"2017-02-28 14:10:20",1.15,"2017-02-28 14:10:00",0,0.031,1.144,0.853,0.931,36.02,14.38,1.098,"2017-02-28 14:10:20",52.24,"2017-02-28 14:10:20",0,0.014,1.196,0.931,35.94, C508^M
160		160, PL20,2017-02-26,12:30:00,0.00, f948
160	XBee Pluviometers	~�8ï¿½�ï¿½�@Ö¼Rï¿½ï¿½ï¿½|SETI160,PL0006, 2017-02-26,12:30:00,0.00,f944
166	Granular Matrix	166,81506,2017-02-20,09:34:00,23.47,"2017-02-2009:34:00",23.47,23.47,"2017-02-20 09:34:00",0,71.62,"2017-02-20 09:34:00",72.1,72.91,"2017-02-20,09:34:00",1.02,"2017-02-20 09:34:00",0,0.001,1.171,0.061,0,D971

a. Id 101 is a message containing daily readings from different meteorological station sensors. MOM versatility allows the middleware to process the 36 parameters included in this type of frame through the interpreters.

b. The first id 160 corresponds to rainfall variables sent by a device through a GSM (Global System for Mobile communications) network. The message identifier (frame) 160 can be observed.

c. The second id 160, shows a frame that also contains daily readings of rain gauge variables sent through an XBee network. Parameters 1 and 2 are discarded and represent the header of a Xbee frame and its delimiter. Parameter 3 is the identifier of the frame type. Parameter 4 is the name or id assigned to a sensor (rain gauge). Parameters 5, 6 correspond to date and time of the data read by the sensor. Finally, there is the checksum parameter, which represents the sum of the number of bytes of parameters 3 and 4; in other words, 9 + N bytes (See Table III).

d. Id 166 corresponds to a frame that contains readings of granular matrix sensors. With this data, the matric potential or volumetric content of soil water is calculated.

TABLE III. FRAME FORMAT IN XBEE NETWORK.

Order	Parameter		
	Name	# Bytes	Hexadecimal representation
1	XBee Heading		
2	Delimiter		
3	Frame type	3	31 36 30
4	Sensor ID	6	50 4c 30 30 30 36
5	Reading	N	XX XX …. XX
6	Checksum	1	XX

The middleware also validates the quality of the frame through the checksum processing. This task is performed by the interpreter, the quality of the data is validated by the user by adding or subtracting the checksum parameters.

IV. IMPLEMENTATION

A case study was carried out on the irrigation planning in a large-scale sugarcane crop. The system processes requests and frames with measurements of water conditions in the soil. With these variables, the soil matric potential (PMS from its Spanish initials) is calculated. PSM indicates the total amount of water available to be absorbed by the plant. In this scenario, the behavior of water in the soil of a sugarcane crop is evaluated.

The crop has loamy soil, is located at 1200 meters above sea level, having the coordinates 3.4281 (latitude) and -76.3071 (longitude). The frames with the information are sent through a hybrid network that combines the DigiMesh topology (ZigBee network) and a point-to-point network (Mobile Cellular Network) (see Fig. 4). This configuration was selected due to project constraints in spite of the fact that the middleware has a sequence of TCP sockets.

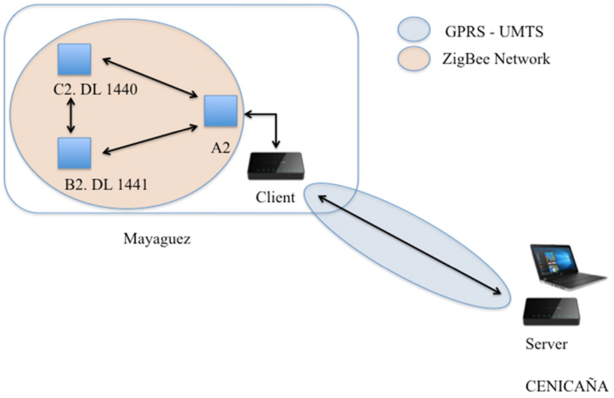

Figure 4. Hybrid network implemented in the monitoring site [47]

Fig. 4 shows the union of two topologies and their point of convergence as follows.

In this topology, the nodes are connected as a mesh network. The second topology (highlighted in blue) supports a client/server model. In the latter, the client modem is connected in series with the XBee® modules and the server modem to the remote computer terminal in CENICAÑA. To communicate the server modem and the client, these devices must be located in an area covered by a mobile cellular telephone network. To configure the network, all devices (XBee® modules, dataloggers, and modems) share a common language (protocol). XBee® modules are assembled on serial cards and provide a serial interface. These modules are connected to the data loggers and the modem through its RS 232 serial port.

TABLE IV. DEPTH AND GROUPING OF GRANULAR MATRIX SENSORS

Sensor MG	Group	Depth
1, 2, 3	P1	30 cms
4, 5, 6	P2	45 cms

Table IV presents the granular matrix sensors and the depth in which they were deployed.

Fig. 5 shows the behavior of the matric potential for each sensor of a granular matrix of the Smart Agricultural Node. This node receives data from six sensors separated into two groups P1 and P2 (at a depth of 30 cm and 45 cm from the soil surface respectively). The sensors are located in a soil with physical characteristics similar to the most of the plants in the crop. The data is sent to the middleware with a format type 166 (table III) at 15-minute intervals. An average of these variables is calculated by the interpreter, according to the frame type. Also, the calibration equation (1) is applied to each reading of each sensor for the calculation of the matric potential [19]. This equation describes the relationships between the resistance in kilo-ohms ($k\Omega$) of the sensors and the soil water potential in kilopascals (kPa). Next, the variables involved in calculating the matric potential are described:

$$PM = \frac{4.093+(3.212*R)}{1-(0.009733*R)-(0.01205*T)} \qquad (1)$$

where PM is the potential of the soil matric in kPa, R is the sensor reading in $k\Omega$, and T is the estimated or measured soil temperature in °C.

The values of matric potential resulting from the application of the calibration equation allow defining mainly two levels of interest to program irrigation works. When the values are lower than -80 kPa, the plants begin to have difficulty extracting water from the soil, and there is a high probability that the crop will suffer from water stress. Conversely, values above 0 KPa, the soil is considered near the saturation point, as shown in Fig. 6.

As the tension increases, the availability of water is lower. In contrast, as the tension decreases, the availability of water is higher, this relationship is known as "change of soil water potential." The available water is between ± -20 kPa and ± -80 kPa; according to criteria established in Cenicaña through experimentation with weighing lysimeters or soil containers described in [20].

Based on these maximum and minimum values, a series of notifications are implemented in the middleware. These messages include push notifications or emails to inform the people in charge of irrigation. The deployment zones of the sensors have approximately a variability of 8 ° C in the soil temperature. Thus, the ranges between ± -20 and ± -80 kPa compensate this temperature difference in the range of 17 and 25 ° C [48].

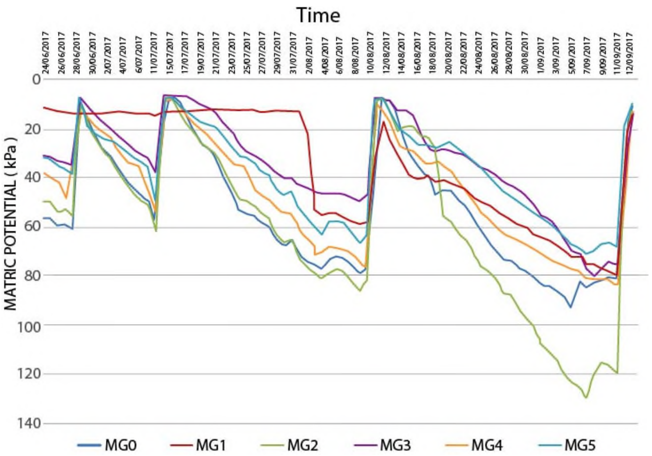

Figure 5. Soil matric potential by sensor

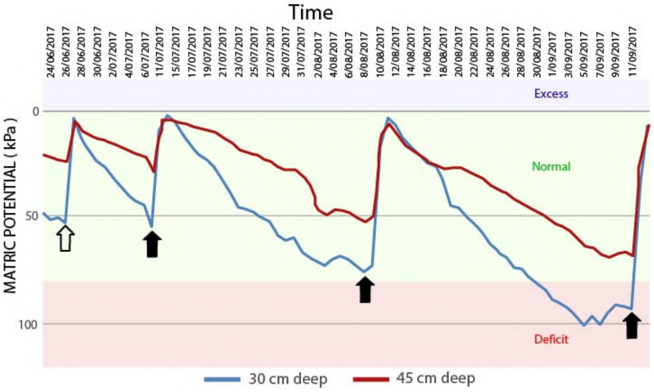

Figure 6. Matric potential vs. time.

Fig. 6 shows curves of average matric potential. It can be seen that on 06/26/2017, just before the irrigation scheduled for 11/07/2017, there was a precipitation event according to rainfall data from the Cenicaña weather station. This rainfall caused an increase in soil humidity (the average matric potential was -53 kPa at a depth of 45 cm, and -24 kPa at 30 cm depth). This situation did not allow observing the change of soil humidity and the behavior of the regular drainage period.

On 08/08/2017 the irrigation began at the "appropriate" moment when the water content of the soil reached a limit close to the deficit. The irrigation scheduled for 11/09/2017 started when the volumetric water content of the soil was in deficit conditions (it can be seen the drainage before irrigation and the recharge period after irrigation). Also, this graph allows visualizing the curve of the readings of sensors at a depth of 30 cm. It can be seen that the decrease in water humidity is faster on the surface due to the heat of the ambient temperature.

In general, the volumetric water content of the soil is very close to the established limits. Water and nutrients remain within the area of highest absorption. Thus, the values stayed close to the field capacity.

With the information of the water content of the soil, people in charge of irrigation planning can make better decisions. Irrigation is carried out only when the soil humidity is at an appropriate level (when), and it can be known when to stop (how much).

V. CONCLUSIONS AND FUTURE WORK

This paper presented the design and implementation of an IoT middleware for the irrigation planning in crops of large expanses of land.

The middleware is composed of four main modules: The CORE allows associating hardware components, software, systems and platforms to establish a machine-to-machine communication. This module focuses on the communication protocols and the subjacent infrastructure. The DAOCore module frees the logic of the middleware (business) from persistence, reducing the complexity of transactions. The INTCore module manages the interpreters that process and parses the messages (frames) received from clients. Finally, the Monitor module sends the commands defined by the users or applications to the Smart Agricultural Nodes. This module generates a bidirectional real-time communication channel for advanced web services and applications and exposes the events that are occurring within the core.

An IoT solution was developed based on the proposed middleware. This solution aims at capturing data from granular matrix sensors, processing this data and calculating the matric potential. This potential allows making decisions on irrigation planning (when and how much).

As future work, it is planned to generate an interpreter automatically when a frame type is defined. Also, the middleware is expected to be used to monitor the matric potential of the soil in the Cauca River Valley. Thus, apart from generating a solution for irrigation, it is expected to be able to acquire a broad knowledge about the behavior of the matric potential in different types of soil and microclimates. This information can be used to analyze other crops and support other widely accepted irrigation models such as the water balance. Finally, a performance evaluation of the multithread processing and communication is envisaged.

REFERENCES

[1] T. Liu, Y. W. Guan, Y. Q. Yan, L. Liu, and Q. C. Deng, “A WSN-Oriented Key Agreement Protocol in Internet of Things,” Appl. Mech. Mater., vol. 401, pp. 1792–1795, 2013.

[2] C. C. Shock, J. M. Barnum, and M. Seddigh, “Calibration of Watermark Soil Moisture Sensors for Irrigation Management,” Int. Irrig. Show, February 1998, pp. 139–146.

[3] F. R. Lamm, J. E. Ayars and F. S. Nakayama, “Microirrigation for Crop Production - Design, Operation, and Management,” vol. 13, Elsevier, 2006.

[4] J. R. Cruz and T. S. Jorge, “El Cenirrómetro: Un Programador Visual de Riegos,(in English: El Cenirrinometer: a visual planner of risks)" Proc. IV Cong. Colombiano de la Asociación de Técnicos de la Caña de Azúcar, Sept.1997, pp. 207-2016.

[5] J. Torres, J. Cruz, and F. Villegas, “Avances técnicos para la programación y el manejo del riego en caña de azúcar (in English: Technical developments in the planning and management of sugar cane irrigation)". Segunda ed. Serie técnica No. 33. Cali. CENICAÑA, 2004.

[6] L. Zotarelli, M. D. Dukes, and K. T. Morgan, “Interpretación del Contenido de la Humedad del Suelo para Determinar Capacidad de Campo y Evitar Riego Excesivo en Suelos Arenosos Utilizando Sensores de Humedad (in English: Understanding of the soil moisture to determine the capacity soil and avoid excesive irrigation in sandy soils through moisture sensors)”. The Institute of Food and Agricultural Sciences (IFAS), pp. 2–5, 2013.

[7] A. C. McCarthy, N. H. Hancock, and S. R. Raine, “Development and simulation of sensor-based irrigation control strategies for cotton using the VARIwise simulation framework,” Comput. Electron. Agric., vol. 101, pp. 148–162, 2014.

[8] T. Ojha, S. Misra, and N. S. Raghuwanshi, “Sensing-cloud: Leveraging the benefits for agricultural applications,” Comput. Electron. Agric., vol. 135, pp. 96–107, 2017.

[9] J. Haule, M. Kisangiri, “Deployment of Wireless Sensor Networks (WSN) in Automated Irrigation Management and Scheduling Systems : a review,” Proc. Science, Computing and Telecommunications (PACT), 2014 Pan African Conference, July 2014, pp. 86–91. doi:

10.1109/SCAT.2014.7055144

[10] A. Ojamaa and K. Duuna, “Assessing the security of Node.js platform,” Proc. *7th Int. Conf. Internet Technol. Secur. Trans.* , 2012, pp. 348–355.

[11] P. Grace, G. S. Blair, and S. Samuel, “ReMMoC: A Reflective Middleware to Support Mobile Client Interoperability,” OTM Confederated International Conferences" On the Move to Meaningful Internet Systems," 2003, pp. 1–18.

[12] OpenIoT Consortium, “Open source solution for the internet of things into the cloud,” January 2012, http://www.openiot.eu [Accessed on: 2018-02-23].

[13] Web NMS, “WebNMS Framework” [Online]. Available: https://www.webnms.com/webnms/. [Accessed on: 2018-02-23].

[14] D. Yan-E, “Design of intelligent agriculture management information system based on IoT,” Proc. - 4th Int. Conf. Intell. Comput. Technol. Autom. ICICTA 2011, vol. 1, March 2011, pp. 1045–1049, doi: 10.1109/ICICTA.2011.262.

[15] W. Zhang and K. M. Hansen, “Semantic web based self-management for a pervasive service middleware,” Proc. 2nd IEEE Int. Conf. Self-Adaptive Self-Organizing Syst. SASO, Oct. 2008, pp. 245–254, doi: 10.1109/SASO.2008.14.

[16] Altair, "Carriots a platform as a Service (PaaS) for the development of IoT and M2M projects", [Online]. Available: https://www.carriots.com/que-es-carriots. [Accessed on: 2018-02-23].

[17] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Cla, “Middleware for internet of things: A survey,” IEEE Internet Things J., vol. 3, no. 1, pp. 70–95, 2016.

[18] B. Tomáš and K. Žáková, “Node.js based remote control of thermo-optical plant,” Proc. 12th Int. Conf. Int. Remote Eng. Virtual Instrum., Feb. 2015, pp. 209-213, doi: 10.1109/REV.2015.7087293

[19] E. P. Eldredge, C. C. Shock, and T. D. Stieber, “Calibration of granular matrix sensors for irrigation management,” Agron. J., vol. 85, no. 6, pp. 1228–1232, 1993.

[20] D. Bahia et al., “Design, installation and calibration of a weighing lysimeter for crop evapotranspiration studies,” vol. 2, no. 2, pp. 77–85, 2013.