A New IoT-based Platform for Greenhouse Crop Production

M. Muñoz, J.L. Guzmán, J.A. Sánchez, F. Rodríguez, M. Torres and M. Berenguel

***Abstract*—This work proposes a cloud solution to build an Internet of Things (IoT) platform applied in a greenhouse crop production context. Real-time and historical data, as well as prediction models, can be accessed by means of RESTful (Representational State Transfer) web services developed for such a purpose. Forecasting is also provided following a Greenhouse Models as a Service (GMaaS) approach. Currently, our GMaaS tool provides forecasting based on computational models developed for inside climate, crop production and irrigation processes. Traditionally, such models are hardcoded in applications or are embedded in software tools to be used as Decision Support Systems (DSS). However, using a GMaaS approach, models are available as RESTful services to be used as needed. In addition, the proposed platform allows users to register new IoT devices and their greenhouse data in the FIWARE platform, providing a cloud scale solution for the case study. RESTful services of the proposed platform are also used by a web application allowing users to interact easily with the system.**

***Index Terms*—Greenhouse Models, IoT, Cloud, DSS.**

I. Introduction

IN the last years, the European Union is boosting the digitalization process in different areas, introducing topics such as IoT, Big Data or Artificial Intelligence in many different contexts (e.g. agriculture, smart cities, smart agrifood, and so on). One example is the IoF2020 project, which introduces and enhances digitalization in several sectors, such as agriculture, livestock, and food in Europe with the aim of significantly improving productivity and sustainability of the system. Within this project, the User Case 4.2 deals with the digitalization of the protected agriculture based on the integration and the use of the data generated by physical and virtual sensors, control loops, networks, models and optimization techniques. FIWARE is another example of the firm commitment of the European Union in the digitalization process.

So, many works are being developed in this context. Smart Agriculture, also-called “thirdgreen revolution”, is presented as the inclusion of services and technologies such as IoT [1], [2], [3], data processing in Big Data [4], cloud computing [5], Farm Management Information Systems (FMIs), and artificial intelligence or Deep Learning [6]. In such a context, Everything as a Service (XaaS) has emerged as a trend, proposing delivering, using an Internet-based access, all the information generated by the integration of technologies, applications or products [7]. Moreover, in this digitalization process, embedded computer-assisted decision support systems are being used as an assistant for farm management. There are many examples of cloud-based decision support toolboxes. For instance, AgroDSS is a cloud-based DSS to combine existing FMIs. This cloud-based toolbox allows farmers to upload data and to use analysis tools and techniques to make future decisions on the system. [8]. Another example is presented in [9], where a cloud-based FMI has been developed and evaluated in a real greenhouse. Recently, solutions proposed by [10], [11] change the vision of these cloud-based technologies proposing an edge-computing and fog-computing platform for traceability and sustainability of the daily farms production to process the data generated at the network edge.

In our context, some challenges are present, such as processing and accessing sensor data, building a cloud solution that can scale up depending on the data requirements, providing an open system so that new and heterogeneous sensors and devices can be easily added, and the possibility to provide greenhouse models as a service.

The aim of this work is to contribute to this digitalization of the agricultural sector, where the implementation of a cloud-based solution for greenhouse crop production is presented. The proposed approach provides different services for economical and environmental benefits of the agricultural activity, and improving the system efficiency by providing suggestions for the use of water, pesticides, fertigation or energy. In this sense, historical and real-time data are available for each of the greenhouses registered in the platform. An open model to allow adding new devices spread over the world has been defined. Weather forecast options are also offered as a service for a 48 hours period. Also, a novel service called GMaaS is integrated into the platform, where greenhouse models are available to estimate indoor climate, crop production and irrigation values. Another advantage is that all these services are available as a REST API for different user needs. In fact, the proposed platform provides a graphical web-based application that is built over this REST API.

Notice that the greenhouse production agrosystem is a very complex process, where physical, chemical and biological processes take place at the same time with different patterns and time scales. For that reason, model-based tools are required as support to understand the dynamics of these systems. Greenhouse climate models, crop growth models and irrigation models have been widely studied in literature [12]. However, most of these models are implemented for research purposes or included as part of specific DSS systems. So, the use of the models by other users (researchers or farmers) is usually

M. Muñoz, J.L. Guzmán, J.A. Sánchez, F. Rodríguez, M. Torres and M. Berenguel are with Deparment of Informatics at the University of Almería, ceiA3, CIESOL, Ctra. Sacramento s/n, 04120 Almería, Spain, +34 950214133, (e-mail:{mmr411,joguzman,jorgesanchez,frrodrig,mtorres,beren}@ual.es).

Manuscript received April 19, 2005; revised August 26, 2015.

limited and complicated.

Thus, the GMaaS option presented in this work provides cloud-based models without any software/device dependence [13]. GMaaS services work through a REST API service implemented in the Matlab Production Server environment [14]. Implemented models can be used for different targets: as simulation tool where the model inputs can be obtained from historical data or from a weather forecast service; as real-time virtual sensor for control/feedback purposes where the model is invoked only one step ahead; and as graphical DSS service from a web-based application. The models implemented in the proposed platform are: inside climate models [15], [12], tomato crop models [16], and water and nitrogen balance model [17].

The remainder of this paper is organized as follows. Section 2 is devoted to describe the proposed IoT system architecture. The different system API services are described in section 3. Then, the different available greenhouse models are described in section 4. Afterwards, examples of the different services are presented in section 5, especially for those based on the GMaaS option. Finally, section 6 is devoted to give some conclusions.

II. IoT-based platform architecture

This section describes the architecture of the proposed IoT-based system which is based on FIWARE. FIWARE is an open source platform funded through a European PPP (Public Private Partnership) project, in which the public and private sectors collaborated to create the Internet of the future. This platform tries to promote the use of new technologies through a collective structure which will contribute to the growth and technological development in Europe. FIWARE is based on a modular architecture, which is supported by a set of GEs (Generic Enablers) that provide a series of functionalities and standards that facilitate the development of intelligent applications. Each of these components may be assembled with other components developed by third parties, thus allowing to accelerate the use of intelligent solutions. These GEs are a set of free and public APIs (Application Programming Interface) based on the formal OMA (Open Mobile Alliance) NGSI (Next Generation Services Interface) [18], [19] specifications with RESTful capabilities, accessible via HTTP. These GEs are separated by chapters depending on the functionality, named as context information management, language interpretation, data analysis, security layers and even web interfaces. FIWARE seeks to become the technological standard that the IoT needs. The core of FIWARE is the GE known as OCB (Orion Context Broker), which manages all the context information produced by the system. It is essential to know the term context because it is the basis of this FIWARE architecture. Context is the name given to all the information that surrounds an ecosystem, which can come from sensor networks, third party applications, public data sources, actuators, among other systems.

Figure 1 shows the cloud architecture of the proposed IoT platform, which is divided into layers: Context Producers, Backend and Frontend. This architecture allows working in a decoupled way. The main advantage of this approach is to allow changes in each layer without affecting the operation in other layers. The aim is to have independent services, even microservices, in each layer. This solution allows to continue developing integration solutions and new improvements to the system. The different layers are described in the following:

1) Layer 0, Context Producers: This is the first layer of the system. It generates all the context information. To generate this information, physical devices must be able to collect information from the environment. These devices consist of a set of microcontrollers and software for updating devices (sensors, actuators, and so on). Sensors only deal with collecting information. Actuators perform an action on their environment. Because of the similarity with the related term in FIWARE, layer names match with Context producer. Notice that Context producer in FIWARE is related to all the information available in the IoT ecosystem from third party systems, sensors, actuators, SCADA systems, and so on.

In our case study, eight greenhouses distributed among the different geographical locations in the province of Almería, Spain, dedicated to tomato crops are used. On the one hand, greenhouses are equipped with heterogeneous meteorological stations. Stations contain many sensors (e.g. CO_2, solar and global radiation, air and soil humidity, electric conductivity, water consumption, air and soil temperature, and etc.). The set of sensors provided by a station depends on its model and its manufacturer. On the other hand, greenhouses may also be equipped with other sensors and actuators connected to data acquisition systems. Figure 1 illustrates how systems and devices installed on greenhouses are connected to the Internet sending data to the next layer, the backend. Meterological stations, data acquisition systems, and SCADA computers provide their data through RESTful services. However, data provided are heterogeneous following no standards.

2) Layer 1, Backend: Located in the center of Figure 1, it deals with data extraction from Layer 0, data processing, databases and REST services. Frontend clients may be created over this layer. Below, the main components are described:

- *IoT Agent and Cron process*: This service deals with extracting context information, transforming and sending it to the IoT system. There are two services, named as IoT Agent and Cron process. IoT Agent transforms sensor values to the NGSI standard of FIWARE in order to solve the problem of interoperability between sensors. The service gets sensor data from different data sources (Layer 0), such as REST services, IoT stations, intelligent sensors and SCADA systems. The IoT Agent service is a developer-enabler for FIWARE which translates different communication protocols to the FIWARE standard. In the figure, it translates data from the meteorological stations to the NGSI standard in order to be sent to the Orion Context Broker (OCB). The another service is a Linux Cron process that periodically obtains data from REST services of the meteorological and SCADA systems, and transform and send them to OCB.
- Orion Context Broker (OCB): It manages context information generated by IoT stations [20]. The OCB data model is based on entities, attributes and metadata. In

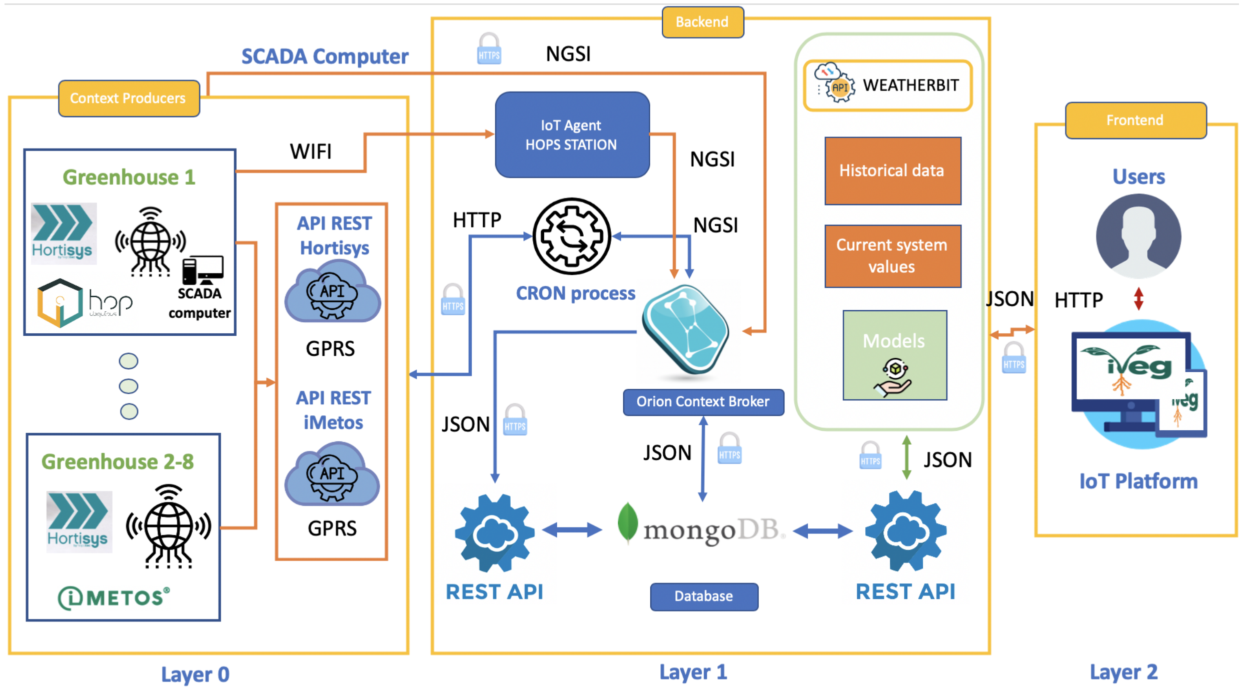


Fig. 1. IoT-based system architecture. The different system layers are depicted, with information about the data flow, communication protocols, data sources, and users.

our context, there is an entity type for each greenhouse station. FIWARE-Service and FIWARE-Service-Path headers are used to identify greenhouse and station names. Each entity is subscribed to the REST API service (see Figure 2), which stores the data in a database. So, whenever an attribute value changes, the new values are notified to the system, proceeding to store them in the database. Additional technical information about the data architecture can be found in [21].

- REST API: The backend layer is accessed using a REST API. Two REST APIs are available. The first one takes care of data persistence, collecting incoming notifications from OCB. The other one deals with the rest of requested operations by the client. Both APIs are based on Node.js. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across the distributed device. Considering that the backend must the scalable, a horizontal scalable database is needed. Aggregation and sharding features are also needed. So, MongoDB was selected attending to those requirements. Backend requests are secured using JWT (JSON Web Token). JWT security is based on a token. When users try to sign in, they send login data to the server (Backend), which generates a JWT and sends it to the client. Every client request will need this JWT. Otherwise, requests are not attended. In addition, custom frontends may be developed over it, considering that Backend is based on a set of REST APIs.
- DSS: This component is responsible for performing system control operations. It is based on Matlab Production Server. It provides a REST API that gets parameters and methods needed to perform control operations. After processing requests, results are returned to the client in JSON format. This new approach of models as a service was proposed in [13].
- Database: The proposed architecture is supported by MongoDB. It is a non-relational document-oriented database. Documents are stored in BSON, a binary representation of JSON. Highlight features are related to performance, replicasets to provide high availability, and sharding to allow horizontal scaling.

3) Layer 2, Frontend: A Single Page Application (SPA), iVeg, has been developed. iVeg has been developed under the project IoF2020 on digitalization in agriculture [22], [21]. The original iVeg application integrated heterogeneous data sources from different service providers at different time scales from sensors and actuators. The current version has been connected to the proposed platform, using Backend and Context Producers provided by Layer 1 and Layer 0, respectively. End users use the application to make requests querying the services provided by Layer 1 in a transparent way. Section 5 shows an example illustrating this layer.

Thus, users/farmers may register their greenhouse devices in the proposed platform. Sensors and actuators data are stored in the database by means of a request to the REST API. Measures provided by the different devices are stored using a variable sampling time within 1-10 minutes, depending on the variable to be saved. Stored data are used to calibrate and to integrate specific models for inside climate, crop production, and irrigation into the platform, which are available as API services.

III. System Services

In this section, the available services are described. Services are classified into two groups: data services and greenhouse models as a service. Figure 2 depicts available data sources, interfaces, and services. The different available services are numbered: historical data (1), real-time data (2), weather forecast (3), and models (4). These services may be used by means of a REST API. JSON is used for information exchange. End users can use these services using the iVeg frontend described at the end of the previous section, or using the REST API directly.

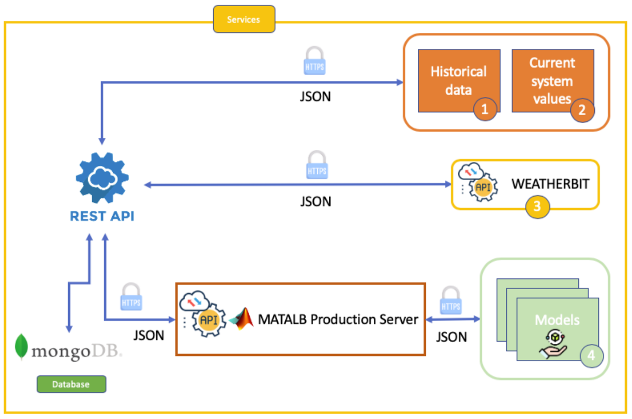


Fig. 2. System services. The different data sources, users and services are represented.

A. Data services

The proposed architecture (Figure 2) provides three sorts of data services. The first data service (1) deals with historical data. It allows users and services to request data stored in the database. Request parameters include *sensor id, station id, start date* and *end date*. The rest of the parameters are available in Table I.

The second data service (2) allows to recover last sensor data as a real time measure. Request parameters are *sensor id* and *station id*. Services 1 and 2 have been developed using the Node.js framework and use JSON Web Tokens (JWT). Tokens are sent in the request header. The system will check the validity and the expiration date of tokens. Additional details of the services can be found in [22], [21].

The third data service is an outdoor weather forecast service (3). This service is based on the REST API service provided by Weatherbit [23], which allows to obtain weather forecasts and historical data in different geographical locations. For this project, the free license option is used, that allows to request weather forecasts 48 hours ahead with an hourly sampling period. Temperature, humidity, wind speed and solar radiation forecasts are used by the proposed IoT platform. Request parameters are *latitude, longitude, language, key* and *number of hours*, as shown in Table I. The service is useful for farmers, since it helps them to make future decisions on the greenhouses according to weather estimations for the next 48 hours. Moreover, it is an essential element for the GMasS service, which is described in the following section.

B. Greenhouse Models as a Service

This is core service of the proposed IoT platform, and can be considered as the main service contribution of this work (service number 4 in Figure 2). As commented above, when a greenhouse is registered in the system, specific models for inside climate, crop production, and irrigation are implemented in the platform as particular services. These model-based services provide the user a DSS tool to obtain forecast about indoor climate conditions, crop production, or irrigation needs, among others. The models are implemented in M code using Matlab and are embedded as services in the Matlab Production Server environment. So, the models can be requested through a service obtaining a response in JSON format [7]. The main advantages of this architecture are:

- Versatility by means of parametrized requests.
- Specific software or implementation is not required to interact with the system.
- Models are available as a cloud service.
- Model coefficients can be updated in a straightforward way according to the user requirements.

The models can be requested for different purposes such as are summarized in the following.

1) GMaaS as a graphical DSS: As discussed previously, users can access into the system by using iVeg as graphical frontend. In this case, users/farmers request this service through a web-based application, which is responsible to make all requests to the system automatically. Once the model outputs are obtained, the information is visualized in a graphic form in the aforementioned application. This option is really powerful for farmers, since they can use the proposed model-based services to make predictions and estimations about climate, production, and irrigation of their greenhouses to be used as DSS. In fact, when the models for climate and irrigation are requested, the system provides suggestions for the control system inputs such as set-point temperature, irrigation volume, irrigation time, or irrigation nitrogen for the next 48 hours.

2) GMaaS for research purposes: A second way to request the GMaaS is by directly used the REST API of the system. This option allows the user to invoke the model for simulation purposes or model predictions from any programming environment or software tool, what it is very useful for research purposes. The user must provide the model inputs (control signals and disturbances) for a certain prediction/simulation horizon. Then, the model is run using these inputs and the model outputs are sent to the user in JSON format (that the user should manage from any REST client).

This request is made by using the HTTP protocol and using the parameters and methods described in Table I for each model. Once the request reaches the models, the Matlab Production Server performs the necessary calculations included in the models and a collection of data vectors with the simulation results are returned to the user in JSON format. When the user requests to use the Weatherbit REST API, geographical information about the greenhouse location is required. Then, this query provides a 48-hour prediction for the climate model inputs, and these data are parsed in vector form and sent as a request to the REST API into the GMaaS option. Next, the

Service	HTTP Method	Operation Name	User parameters
Register new IoT station	POST	/v2/entities/	station id, station type, attributes, type, value, metadata (...)
Historical data	GET	/getDataSensorGreenhouseAnalyticsFechasV2	sensor id, station id, start date, end date
Current system values	GET	/getDataSensorGreenhouseLastDataV2	sensor id, station id
Weather forecasts	GET	/forecast/hourly?	lat, lon, lang, key, hours
Climate model	POST	/f_climate/f_climate	[[indoor temp, indoor rel. hum., floor temp, lay], [outside temp forecast], [outside wind speed forecast], [outside rad forecast], [outside humidity forecast]]
Production model	POST	/f_tomgro/f_tomgro	[sensor id, station id, start date, present date]
Irrigation model	POST	/f_irrigation/f_irrigation	[sensor id, station id, start date, present date, lat, long]

TABLE I
IOT-BASED PLATFORM SERVICES

model is simulated using these weather forecast data and the simulation results for the next 2 days are returned to the user in JSON format. Therefore, short- or long-term simulations can be done with this service for any of the available models.

IV. MODELS FOR THE GMAAS OPTION

Such as described above, the GMaaS option is the most relevant service of the proposed IoT-based platform. Thus, this section briefly describes the main features of the implemented models for greenhouse inside climate [12], crop production [16], and irrigation [17]. All these models can be requested as services, where the different service options available through the REST API are summarized in Table I.

1) Climate model: The greenhouse climate model implemented in the platform is developed as a combination of individual models to estimate the inside climate conditions for temperature, global radiation, CO_2 and relative humidity [12]. These models use external predictions to estimate the different variables as a complex process that depends on the particular characteristics of these systems, with different climatic actuators, cover materials, structures, and crops, and the mass and energy balances produced among the different elements (cover, soil, internal and external air and crops). The number of equations to be solved depends on the facilities installed and the exchange with the different components.

The temperature model is based on the energy balance produced between the inside air and the different elements that take part in the crop production processes in protected agriculture. The temperature general model is divided into seven differential equations for a greenhouse with ventilation, shadow net and aerothermal heating as those used for this work. This subsystem includes: the convective flux of the internal air with the plastic cover, shortwave radiation absorption, the convective flux with the soil surface, the latent heat effect from crop transpiration, the convective flux lost when the greenhouse is closed, heat fluxes of the different layers inside the soil, and the latent heat effect from the soil evaporation. Furthermore, depending on the systems installed, the general equation of the air temperature can include other like: the heat lost by natural ventilation and the heat lost by infiltration losses, the heat fluxes with the pipe-based heating systems, the heat fluxes with the aerothermal heating system, and the attenuation of the low wave radiation absorbed due to the shadow net (from radiation model).

On the other hand, the air relative humidity model is based on the mass exchange (absolute humidity) among the different elements of the system. In this case, the general model is composed by only three differential equations: water condensation on the plastic cover, water released by crop transpiration, and crop released by soil evaporation. For this model, only water losses by natural ventilation, water released by humidification, and water condensed by the dehumidification system has influence on the relative humidity.

Moreover, CO_2 dynamic prediction is also included and depends on the carbon released by different methods of enrichment (pure CO_2 and different inputs combustion), soil degradation and the crop respiration. CO_2 losses are due to photosynthesis consumption for the biomass growth and by natural ventilation.

Finally, inside global radiation is estimated through the characteristics of the plastic cover and modifications of the plastic transmissibility with the shadow net or the whitening process.

Thus, the climate model is described by following general differential equation:

$$\frac{dX_{cl}}{dt} = f_{cl}(X_{cl}, U_{cl}, D_{cl}, V_{cl}, C_{cl}, t) \tag{1}$$

where $X_{cl} = X_{cl}(t)$ is a vector of greenhouse climate state variables (inside global and PAR radiation, air temperature, humidity, and CO_2 concentration), $U_{cl} = U_{cl}(t)$ describes the control variables (natural vents, heating system, humidification, dehumidification, screens, and CO_2 enrichment), $D_{cl} = D_{cl}(t)$ is a vector of disturbances (leaf area index of the crop, soil surface temperature, outside radiation, air temperature, humidity and CO_2 concentration, rain, and wind speed and direction), $V_{cl} = V_{cl}(t)$ represents the system variables related to physical processes of the heat and mass balances (convective, conduction, thermal radiation, and latent heat), C_{cl} is a vector of system constants related to the climate processes, t is the time, and $f_{cl} = f_{cl}(t)$ is a nonlinear set of functions based on mass and heat transfer balances. More details about the model can be found in [12].

2) Production model: Currently, the production model implemented in the IoT-based platform is based on tomato crop, although extensions to other crop models can be easily included. Therefore, the *Tomgro* crop model is the one available through the GMaaS system. The main state variables in this model are: total dry weight, leaf area index (LAI), dry matter of fruits, and dry matter of mature fruits, and the number of nodes. More information about the equations and the model can be found in [16] and [24].

The main features of this model are the following. The number of nodes is a function of the speed of nodes formation, which depends on the greenhouse temperature. The LAI calculation includes daily average temperature, empirical coefficients and plant density. The total dry matter depends on the crop growth rate. The growth is mainly calculated as a function of the photosynthesis rate minus the crop respiration, and modulated by a function of the dry matter distribution to the roots, which depends on the number of nodes. The photosynthesis rate calculation is based on greenhouse temperature, photosynthetically active radiation (PAR) radiation, CO_2, and LAI. On the other hand, the respiration term is obtained based on the greenhouse temperature and the total dry matter. The dry matter of fruits includes the phenomena of fruit allocation and transition from vegetative to reproductive growth stages. Moreover, it considers the effect of the average greenhouse temperature on the distribution between vegetative and reproductive growth.

Finally, the dry matter of mature fruits is based on the assumption that the mature fruits are immediately harvested. So, it is calculated using a function on the temperature effect over the fruit ripening. This calculation is activated when a specific number of nodes is reached.

As a summary, the tomato crop growth model can be described by the following differential equation:

$$\frac{dX_{gr}}{dt} = f_{gr}(X_{gr}, U_{gr}, D_{gr}, V_{gr}, C_{gr}, t) \tag{2}$$

where $X_{gr} = X_{gr}(t)$ is a vector of crop growth state variables (total dry weight, leaf area index, dry matter of fruits, and dry mater of mature fruits, and number of nodes), $U_{gr} = U_{gr}(t)$ is the control variables of crop growth (air temperature, CO_2 concentration and PAR radiation), $D_{gr} = D_{gr}(t)$ is the vector of disturbances (cultural labors, electrical conductivity, diseases, and air humidity,) $V_{gr} = V_{gr}(t)$ represents the system variables related with physiological processes (photosynthesis and respiration), C_{gr} is a vector of constants, t the time, and $f_{gr} = f_{gr}(t)$ is a nonlinear set of functions based on the basic physiological processes of the plants. More details about the model can be found in [12], [16].

3) Irrigation model: The fertigation model uses the temperature, humidity and growth predictions from the rest of the models in the GMaaS to estimate the crop irrigation and nitrogen necessities [17]. The system allows to estimate the daily needs of nitrogen as nutrient of the plant and the water needs for the nutrient solution applied through fertigation and drip irrigation in greenhouse crops. The solution can be used in soil or substrate crops. The nitrogen necessities are estimated from the crop nitrogen extraction, nitrogen introduced by irrigation, the mineral present in the soil at the beginning of the season and the mineralized nitrogen from manure and organic matter from the soil. Irrigation needs are calculated from crop evapotranspiration using the Leaf Area Index (LAI) from the crop growth model and the inside climate model available described above. Thus, the main differential equation for the irrigation model can be expressed as follows:

$$\frac{dX_{ir}}{dt} = f_{ir}(X_{ir}, U_{ir}, D_{ir}, V_{ir}, C_{ir}, t) \tag{3}$$

where $X_{ir} = X_{ir}(t)$ is a double state variable vector (crop evapotranspiration and Nitrogen extraction), $U_{ir} = U_{ir}(t)$ is the vector of control variables (irrigation duration and crop Nitrogen necessities), $D_{ir} = D_{ir}(t)$ is the vector of disturbances composed by the inside climate variables (air temperature, humidity, and PAR radiation), crop state (total dry matter and leaf area index), and cultural labors (manure), $V_{ir} = V_{ir}(t)$ are system variables related with physical, chemical, and physiological processes (Nitrogen available in soil, Nitrogen demand, sensible and latent heat), C_{ir} is a vector of constants, t the time, and $f_{ir} = f_{ir}(t)$ is a nonlinear function based on water and nitrogen balances on the soil-plant atmosphere. More detailed about the model can be found in [17].

V. Examples of the proposed services

This section shows several examples of different IoT-based services, especially for the GMaaS option. The web-based application iVeg, described in section II, is used to show the main capabilities of the system. Services are available and integrated in the application as icons at the top, as shown in Figures 3-6. Examples are shown in the following subsections.

A. Register a new IoT station

This section describes how a new IoT station can be registered in the system. A short code description of the request body is shown in the following:

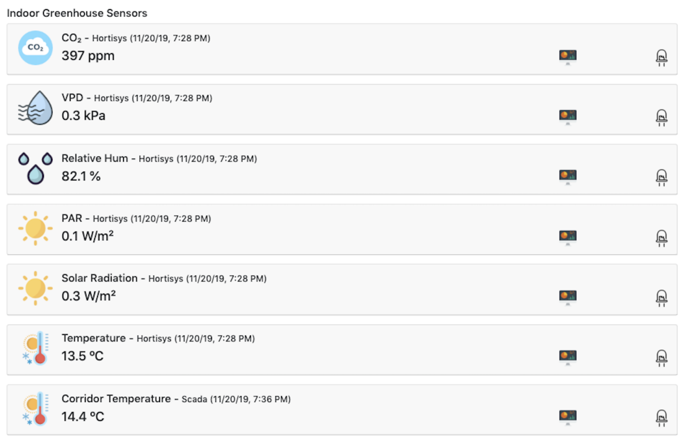
```
{
  "id": "Station_greenhouse2",
  "type": "Station_x",
  "Temperature": {
            "type": "float",
            "value": "24",
            "metadata": {
                "id": {
                    "type": "String",
                    "value": "id_1"
                },
                "timestamp": {
                    "type": "Date",
                    "value": "06/11/2019
                                 21:44:20"
                }
            }
        },
    "Humidity": {
                ...
            }
}
```

For such a purpose, the service “Register new IoT station” (see Table I) must be called. A POST request is sent using the variables described in the table. The NGSI structure defined by FIWARE in OCB must be used. In addition, different types of attributes, such as sensor name, manufacturer and model, location, sensor type, and so on, may be added as metadata.

B. Historical data and current system values

When using the historical data service, the user selects several sensors from a list and a range of dates to consult. The answer is visualized in a single graph combined with the values of the selected greenhouse sensors. On the other hand, the services for the current values of each sensor are represented in the application in the form of a card with the real-time values of the system. An example is shown in Figure 3.

The detailed information for the request of these services were described in Section 2.A and are accessible by using the second/third option in Table I, respectively.


Indoor Greenhouse Sensors

CO_2 - Hortisys (11/20/19, 7:28 PM)
397 ppm

VPD - Hortisys (11/20/19, 7:28 PM)
0.3 kPa

Relative Hum - Hortisys (11/20/19, 7:28 PM)
82.1 %

PAR - Hortisys (11/20/19, 7:28 PM)
0.1 W/m²

Solar Radiation - Hortisys (11/20/19, 7:28 PM)
0.3 W/m²

Temperature - Hortisys (11/20/19, 7:28 PM)
13.5 ºC

Corridor Temperature - Scada (11/20/19, 7:36 PM)
14.4 ºC

Fig. 3. Example of service current value using the iVeg application.

C. GMaaS examples

This section is devoted to show how the different greenhouse models are requested and how the obtained information is visualized.

1) Climate model: This section describes the use of the internal climate model of the greenhouse described in section III, which is one of the services available through the GMaaS option. In this case, the GMaaS has a REST API service that allows the user to know the future climatic conditions inside the greenhouse at 48 hours in an hourly range of one hour. The main state variables (see Table I) in this model are the current sensor values and the future variable values. This request makes use of the services detailed above: current system values and weather forecasts. These types of variables are included in array of arrays as shown in Table I. The first array includes 5 arrays with each of the parameters necessary for the model operation (see Table I). The parameters of the first position in the array are the inside greenhouse variables that are obtained using the current system values service, such as: indoor temperature, indoor humidity, soil temperature, and LAI. The rest of the arrays are formed by 48 future values that are obtained by using the weather forecast service: outside temperature, outside wind speed, outside radiation and humidity. Once the request has been sent to the GMaaS service, two different simulations are carried out 48 hours ahead, one considering the ventilation totally open and other one with the ventilation totally closed. These calculations allow to know the internal conditions of the greenhouse climate according to the weather forecast and for the two extreme cases of the actuator states. Then, the system returns the response to the user in JSON format structured in 7 arrays with the following information: time in hours, open ventilation temperature, open ventilation humidity, closed ventilation temperature, closed ventilation humidity, and inside radiation.

When the request is done using iVeg, the user does not need to perform any operation, since the application is responsible for recovering all the necessary information from the database and calling each of the necessary services. The resulting answer is drawn in 3 graphs combining the closed and open ventilation values with temperature, humidity, and radiation. Figure 4 shows an example of this service, where the bands for minimum and maximum apertures in ventilation can be observed for the two simulated days. Notice that this information is really useful to help the farmers in the day-to-day climate control decisions, especially to provide advices about the control system set-points.

2) Production model: This model as a service allows the user to know the production of the greenhouse from the beginning of the campaign until the moment the service is requested. The GMaaS encapsulated production model has a REST API service invoked through the HTTP protocol. The input variables required to run this service are represented in Table I. This model is fed from the services and models described above for data history information and current sensor values. The response of this model returns five variables in JSON format to the user: time, fruits dry weight, total dry weight, mature fruits dry weight, and LAI.

When the web application is used, the user accesses the models tab and selects the *Production Model* option. Then, only the start date of the campaign must be selected, and the platform is in charge of recovering all the sensor information and the end date. The answer is visualized in the form of four graphs with the values of fruits dry weight, total dry weight, mature fruits dry weight, and LAI (see Figure 5 for an example).

3) Irrigation model: This section shows how the indoor greenhouse irrigation model works, which is one of the services available through the GMaaS system. It allows the user to know the future water needs inside the greenhouse, recommending irrigation instructions for the next 48 hours. The model is called by using the input data shown in Table I. The main state variables needed in this model are id sensor, id station, campaign start date, current date, latitude and longitude. This service makes use of all the services already described and as a result, the model provides estimations and suggestions in JSON format for irrigation time, irrigation volume, and nitrogen requirements since the beginning of the campaign to 48 hours from the consultation time. This service can be used in the web application through the model tab by choosing the *Irrigation Model* option. Then, the campaign start date is selected, and in this way, the system responds to the request by drawing three graphs with the information about irrigation time, volume of irrigation, and nitrogen requirements. Moreover, a table with future irrigation instructions at 24 and 48 hours is generated. Figure 6 shows an example where all

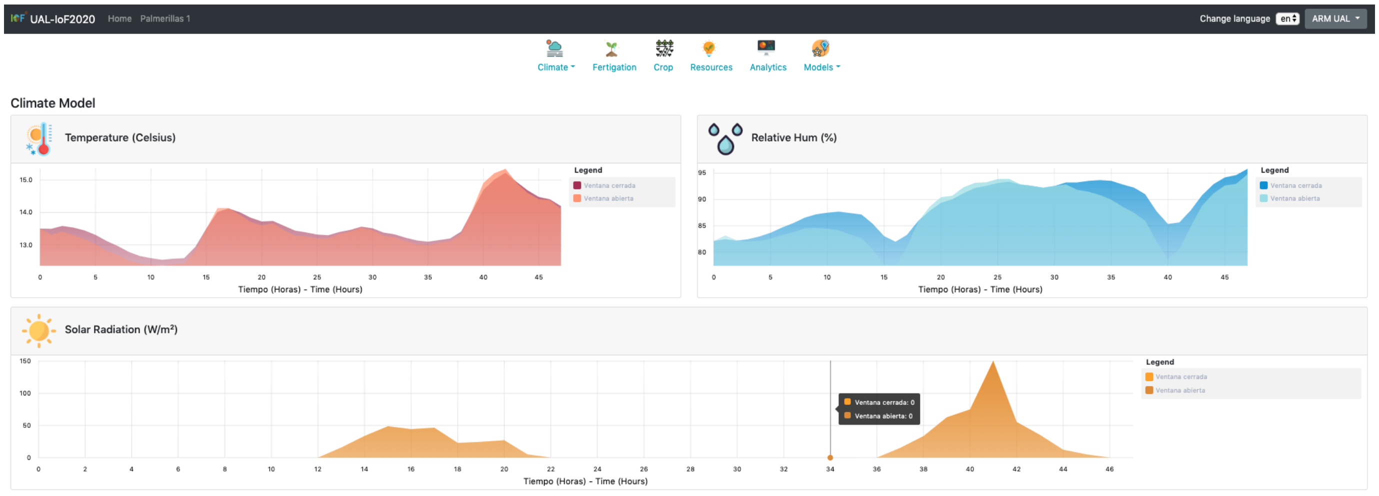


Fig. 4. Example of the climate service using the iVeg application.

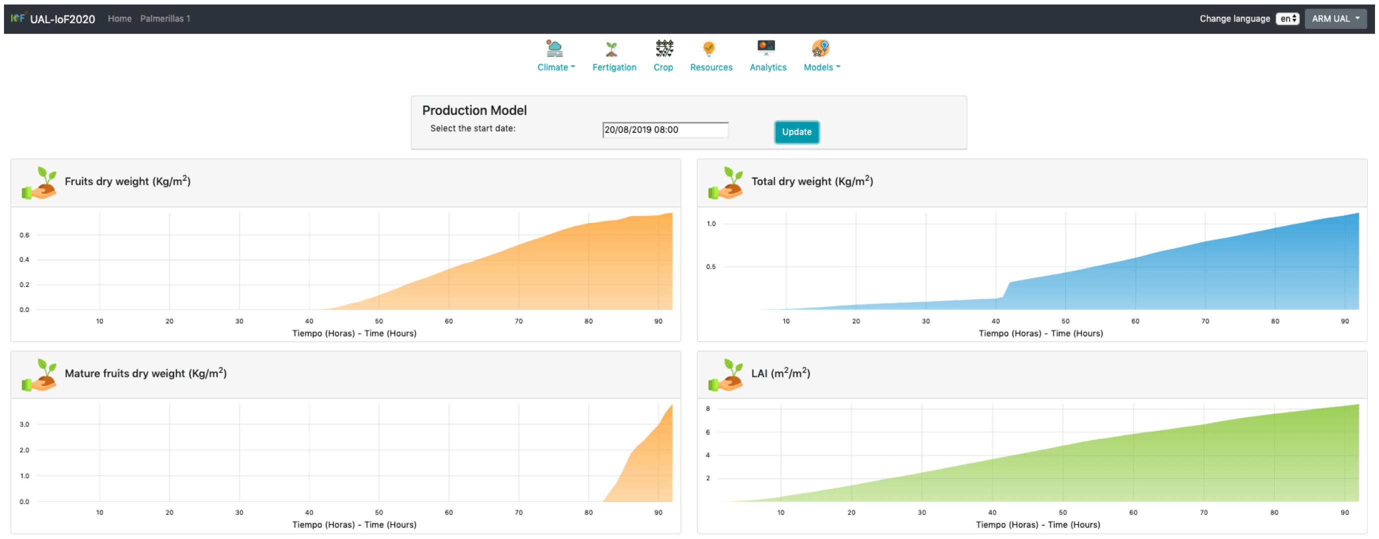


Fig. 5. Example of the production service using the iVeg application.

this information is shown.

VI. Conclusion

A cloud-based solution to provide services for the greenhouse crop production problem has been presented in this work. The services available in the proposed IoT-based system are: historical data, current values, weather forecasts, climate model, tomato production model, and irrigation model; which are available through the resulting REST API service. This API allows the versatility to access the services through two modalities: the iVeg web-based application developed under the framework of the IoF2020 project (use case 4.2 vegetables) or by directly using the REST API. Notice that this last option is really useful for research purposes, since any user can request data or models from any REST client. Therefore, the data and model-based services can be used for simulation/study purposes or for industrial purposes as a DSS by farmers/companies.

Acknowledgment

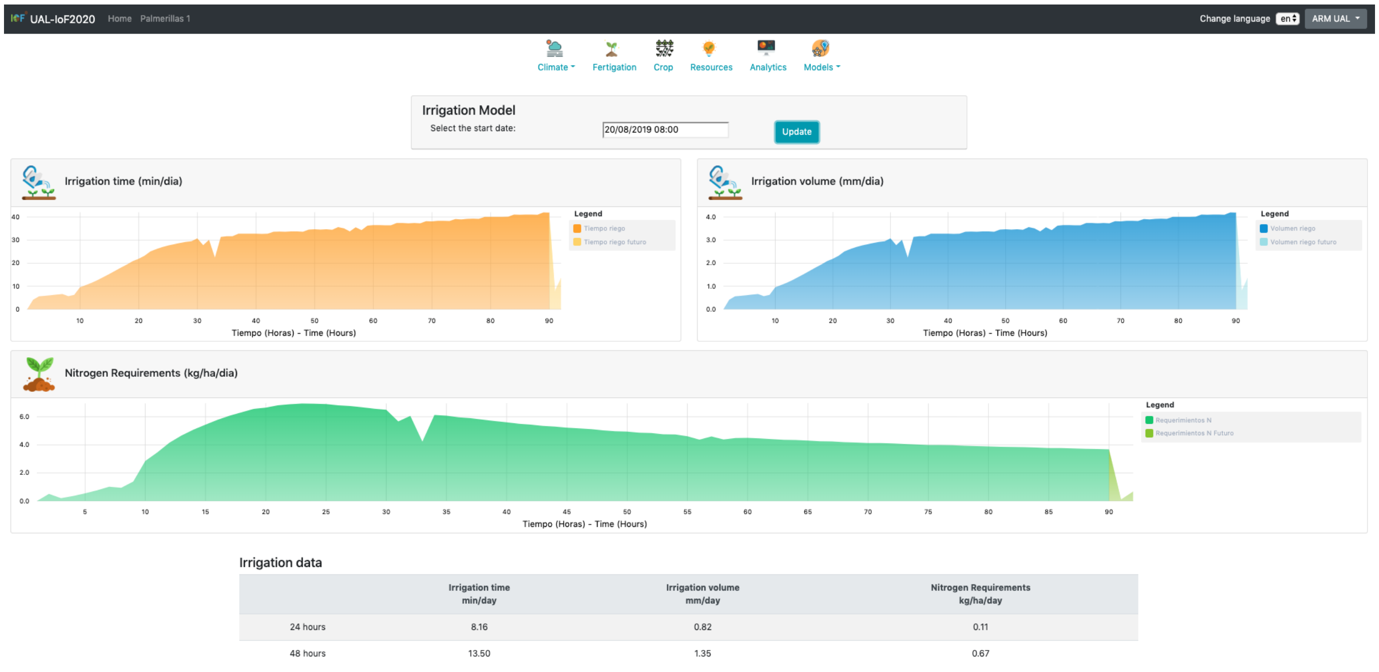
This work has been developed within the following projects: DPI2017-84259-C2-1-R financed by the Spanish Ministry of Science and Innovation and EU-ERDF funds, and the European Union's Horizon 2020 Research and Innovation Program under Grant Agreement No. 731884 - Project IoF2020-Internet of Food and Farm 2020.

References

[1] J. Liu and J. P. Tao, "Research and application of agricultural greenhouse intelligence platform based on IoT (Internet of Things) and cloud computing," *International Journal of Simulation – Systems, Science & Technology*, vol. 17, no. 5, pp. 8.1–8.5, 2016.

[2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, sep 2013.

[3] O. Elijah, T. A. Rahman, I. Orikumhi, C. Y. Leow, and M. N. Hindia, "An overview of internet of things (IoT) and data analytics in agriculture: Benefits and challenges," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3758–3773, oct 2018.



Irrigation data

	Irrigation time min/day	Irrigation volume mm/day	Nitrogen Requirements kg/ha/day
24 hours	8.16	0.82	0.11
48 hours	13.50	1.35	0.67

Fig. 6. Example of the irrigation service using the iVeg application.

[4] A. Kamilaris, A. Kartakoullis, and F. X. Prenafeta-Boldú, "A review on the practice of big data analysis in agriculture," *Computers and Electronics in Agriculture*, vol. 143, pp. 23–37, 2017.

[5] R. Shamshiri, F. Kalantari, K. C. Ting, K. R. Thorp, I. A. Hameed, C. Weltzien, D. Ahmad, and Z. M. Shad, "Advances in greenhouse automation and controlled environment agriculture: A transition to plant factories and urban agriculture," *International Journal of Agricultural and Biological Engineering*, vol. 11, no. 1, pp. 1–22, jan 2018.

[6] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and Electronics in Agriculture*, vol. 147, pp. 70–90, apr 2018.

[7] Y. Duan, G. Fu, N. Zhou, X. Sun, N. C. Narendra, and B. Hu, "Everything as a Service (XaaS) on the Cloud: Origins, Current and Future Trends," in *IEEE 8th International Conference on Cloud Computing*, 2015, pp. 621–628, noida, India.

[8] R. Rupnik, M. Kukar, P. Vračar, D. Košir, D. Pevec, and Z. Bosnić, "AgroDSS: A decision support system for agriculture and farming," *Computers and Electronics in Agriculture*, vol. 161, pp. 260–271, 2019.

[9] A. Kaloxylos, A. Groumas, V. Sarris, L. Katsikas, P. Magdalinos, E. Antoniou, Z. Politopoulou, S. Wolfert, C. Brewster, R. Eigenmann, and C. Maestre Terol, "A cloud-based Farm Management System: Architecture and implementation," *Computers and Electronics in Agriculture*, vol. 100, pp. 168–179, jan 2014.

[10] R. S. Alonso, I. Sittón-Candanedo, Ó. García, J. Prieto, and S. Rodríguez-González, "An intelligent edge-IoT platform for monitoring livestock and crops in a dairy farming scenario," *Ad Hoc Networks*, p. 102047, 2019.

[11] N. Ahmed, D. De, and I. Hussain, "Internet of things (IoT) for smart precision agriculture and farming in rural areas," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4890–4899, dec 2018.

[12] F. Rodríguez, M. Berenguel, J. L. Guzmán, and A. Ramírez-Arias, *Modeling and control of greenhouse crop growth*. Springer, 2015.

[13] M. Muñoz Rodríguez, J. L. Guzmán, J. A. Sánchez-Molina, F. Rodríguez, and M. Torres, "Greenhouse Models as a Service (GMaaS) for Simulation and Control," in *6th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture*, December 2019, Sydeny, Australia.

[14] I. The MathWorks, *Matlab User Guide*. https://es.mathworks.com/help/: The MathWorks, Inc., 2019.

[15] J. A. Sánchez-Molina, M. Li, F. Rodríguez, J. L. Guzmán, H. Wang, and X. T. Yang, "Development and test verification of air temperature model for chinese solar and spainish almeria-type greenhouses," *International Journal of Agricultural and Biological Engineering*, vol. 10, no. 4, pp. 66–76, 2017.

[16] J. A. Sánchez-Molina, F. Rodríguez, J. L. Guzmán, and J. A. Ramírez-Arias, "Water content virtual sensor for tomatoes in coconut coir substrate for irrigation control design," *Agricultural Water Management*, vol. 151, pp. 114–125, 2015.

[17] M. Gallardo, F. Arrabal, F. Padilla, M. Peña-Fleitas, and R. Thompson, "Veg syst-dss software to calculate n and irrigation requirements for seven vegetable species grown with fertigation in greenhouses in se spain," *Acta Horticulturae*, vol. 1182, pp. 65–72, 2017.

[18] M. A. Rodriguez, L. Cuenca, and A. Ortiz, "Fiware open source standard platform in smart farming - a review," in *IFIP Advances in Information and Communication Technology*, vol. 534, 2018, pp. 581–589. [Online]. Available: http://link.springer.com/10.1007/978-3-319-99127-6_50

[19] OMA SpecWorks, "Home - OMA SpecWorks." [Online]. Available: https://www.omaspecworks.org/

[20] D. Ferreira, P. Corista, J. Giao, S. Ghimire, J. Sarraipa, and R. Jardim-Goncalves, "Towards smart agriculture using FIWARE enablers," in *2017 International Conference on Engineering, Technology and Innovation: Engineering, Technology and Innovation Management Beyond 2020: New Challenges, New Approaches, ICE/ITMC 2017 - Proceedings*, vol. 2018-Janua. IEEE, jun 2018, pp. 1544–1551. [Online]. Available: http://ieeexplore.ieee.org/document/8280066/

[21] M. Muñoz Rodríguez, J. A. Sánchez-Molina, M. Torres, and M. Berenguel, "IoT-Based APP Architecture for Greenhouse Management," in *EFITA-HAICTA-WCCA Congress*, 2019, rodas, Greece.

[22] F. Rodríguez, M. Berenguel, J. Sánchez-Molina, and M. Muñoz Rodríguez, "Farms, Fogs and Clouds: Data open-architecture for optimal crop growth control for IoF2020 project," in *European Conference on Agricultural Engineering*, July 2018, wageningen, The Netherlands.

[23] I. OpenWeatherMap, *OpenWeather API Guide*. https://openweather.co.uk: OpenWeatherMap, Inc., 2019.

[24] J. Jones, A. Kening, and C. Vallejos, "Reduced state-variable tomato growth model," *Transactions of ASAE*, vol. 42, no. 1, pp. 255–265, 1999.

M. Muñoz is a researcher at the University of Almería, where he graduated as a Technical Engineer in Computer Systems, as well as a Computer Engineer, and where he is currently writing his PhD thesis ”New paradigms for the integration of IoT systems in protected agriculture”. He is developing his research work as a member of the group of Automatic Control, Robotics and Mechatronics (TEP-197) since 2017, specializing in IOT (Internet of objects), BigData, robotics, data acquisition systems, cloud systems, Back-End and Front-End development. So far, he has 6 publications in international and national congresses, having been awarded for the publication “IOT applied to traceability and decision-making for tomato cultivation in greenhouses”. He is currently participating in a European project for the digitalisation of agriculture ”IoF2020”.

J.L. Guzmán received the computer science engineering degree and the European Ph.D. degree (extraordinary doctorate award) from the University of Almería, Spain, in 2002 and 2006, respectively. He is Full Professor of automatic control and systems engineering at the University of Almería. His research interests focus on the fields of control education, model-predictive control techniques, PID control, and robust control, with applications to agricultural processes, solar plants, and biotechnology. He has been a member of the Spanish Association in Automatic Control CEA-IFAC since 2003, the IEEE Control System Society since 2006, and the IFAC Technical Committee on Control Education and the IEEE Technical Committee on System Identification.

J.A. Sánchez is an Associate Professor of Systems Engineering and Automatic Control at UAL, working with the Group of Automatic Control, Robotics and Mechatronics since 2005. He obtained the titles of Agricultural Engineering, Master’s Degree in Industrial Computer Science, Master’s Degree in Industrial and Agri-food Biotechnology and Ph.D. degree in Computer Science (Extraordinary Doctorate Award). His work focuses on the climate, crop growth and irrigation modelling and control in greenhouses, IoT and Big Data applications in agriculture and renewable energies, mobile and grasping robotics. He is currently supervising two Ph.D. thesis. He is enrolled in the Focus Group of Digitalization and Big Data in the agri-food, forestry and rural areas of the Spanish Minister of Agriculture, Spanish Society of Agroengineering (SEA), the International Society for Horticultural Science (ISHS), and the Spanish Association in Automatic Control (CEA).

F. Rodríguez received the M.Sc. degree in Telecommunication Engineering from the Universidad Politécnica de Madrid, Madrid, Spain, and the Ph.D. degree from the University of Almería, Almería, Spain, in 2002. He is currently full Professor of Systems Engineering and Automatic Control with the University of Almería, where he is also a Researcher and a member of the Automatic Control, Mechatronics and Robotics Research Group. He has participated in several Spanish and European research projects, and research and development tasks with many companies. His current research interests include the application of modelling, automatic control, and robotics techniques to dynamic systems and education, concretely energy management and agriculture. Dr. Rodríguez has been a member of the IFAC Technical Committee TC 8.1, Control in Agriculture, since 2005.

M. Torres received the computer science engineering degree from the University of Granada, Spain, and the Ph.D. degree from University of Almería, Spain, in 2002. He is currently Associate Professor in the Informatics Department at the University of Almería. His research interests are focused on the fields of databases and cloud computing.

M. Berenguel received the industrial engineering and Ph.D. (extraordinary doctorate award) degrees from the University of Seville, Spain. He is Full Professor of automatic control and systems engineering and head of the research group “Automatic Control, Robotics and Mechatronics” (arm.ual.es) with the University of Almería, Spain. His research interests include control education, predictive and hierarchical control, and IoT, with applications to solar energy systems, agriculture, and biotechnology. He is Technology Chair of Vegetables Trial in Internet of Food and Farm IoF2020 EU project and also participates in SmartAgriHubs. He is Senior Member of IEEE and In 2019 he became “Honorary Visiting Professor” at University of Brescia, Italy.