




A layered IoT architecture for greenhouse monitoring and remote control

Hassan Ibrahim¹ · Norhan Mostafa¹ · Hassan Halawa¹ · Malak Elsalamouny¹ · Ramez Daoud¹ · Hassanein Amer¹ · Yasmine Adel¹ · Amr Shaarawi² · Ahmed Khattab³  · Hany ElSayed³

© Springer Nature Switzerland AG 2019

Abstract

Wireless Sensor Networks have been often used in the context of Greenhouse architectures. In this paper, an architecture is proposed for two Greenhouses based on Networked Control Systems. This architecture is IoT-based and built on top of switched Ethernet and Wi-Fi. Some sensors in the proposed architecture require a one-second real-time deadline. Riverbed simulations prove that there is zero packet loss and no over-delayed packets. An important contribution of this work is the design of a channel allocation scheme that prevents interference in this relatively large Greenhouse system. Another contribution of this work is the introduction of fault tolerance at the controller level. If one controller fails in one of the Greenhouses, the other controller automatically takes over the entire operation of the two-Greenhouse system. Riverbed simulations again show that this fault-tolerant system does not suffer any packet loss or over-delayed packets. Continuous Time Markov Chains are then developed to calculate the reliability as well as the steady state availability of the two-Greenhouse system. The Coverage parameter is taken into account. Finally, a case study is presented to quantitatively assess the advantage of fault tolerance in terms of downtime reduction; this is expected to be attractive especially in developing countries.

Keywords NCS · IoT · Agriculture · Greenhouse · Ethernet · Fault-tolerance · Reliability

1 Introduction

Greenhouses have a fundamental role in modern agriculture. They are structures that consist of walls and roofs and covered by a transparent material like glass. These structures provide the ability to control different agricultural parameters and conditions so that an increase in plant growth and fruit production can be successfully achieved. Furthermore, they provide the means to overcome harsh conditions and environmental restrictions in plantations and produce crops with very good quality.

Greenhouses are in continuous evolvement starting by deploying different sensor nodes that act as human observers. These sensor nodes replace human

observations and measure the environmental parameters and conditions inside the greenhouse in order to get more precise measurements with the desired sampling frequency. Additionally, they allow gathering and processing of the collected measurements in order to determine the actual status of the greenhouses. Hence, a user interface is set up to display this information to the greenhouse stakeholders [1–3].

Moreover, adding a layer of IoT to this architecture has contributed to the evolvement of agriculture in greenhouses [4]. All the collected information coming from the nodes inside the greenhouse can be analyzed, stored over a cloud, and displayed to the end user from any Internet-enabled device [5–7].

✉ Ahmed Khattab, akhattab@ieee.org | ¹Electronics and Communications Engineering Department, The American University in Cairo (AUC), Cairo, Egypt. ²Physics Department, The American University in Cairo (AUC), Cairo, Egypt. ³Electronics and Electrical Communications Engineering Department, Cairo University, Cairo, Egypt.



The improvement in greenhouses is not only limited to monitoring and reporting any internal change in the environmental parameters and conditions but also allows the remote control of the different conditions inside greenhouses [8, 9]. According to the gathered information coming from the monitoring system, decisions can be taken regarding what should be done inside the greenhouse; the remote control system has the ability to send commands to the greenhouse's different actuators to execute certain actions on irrigation valves and windows, for example. Through the use of microcontrollers, the decisions and commands can be triggered automatically inside the greenhouse when certain environmental condition change or triggered manually by a greenhouse administrator [10–15].

In previous works, however, small to medium systems that depend on wireless sensor networks (WSNs), were studied for proof of concept. In fact, there is no assurance that there is an existing system architecture that can be implemented on a relatively large scale and can guarantee efficiency and reliability. In this paper, the suggested greenhouse architecture is studied as a hierarchical distributed Network Control System (NCS) [7, 8], i.e., an NCS would be suitable since the greenhouse system consists of several sensor nodes, controllers, and actuators which are interconnected by a shared network that is designed for carrying small packets and is required to meet real-time constraints with the least amount of packet loss and high reliability. The NCS is established on top of Ethernet and Wi-Fi as in typical NCSs [16–19]. Unlike the existing literature which typically considers a single greenhouse [1–15], we consider a relatively large system that consists of two greenhouses to study not only the monitoring and control performance but also the fault tolerance capabilities of the system. Therefore, it will be shown how to allocate suitable Wi-Fi channels to enable this relatively large system to meet all NCS real-time constraints such as packet loss and delay [20–22]. Then, fault tolerance is discussed. Since the two identical greenhouses are connected to the same network, it is proven that both can meet system constraints even if an error occurs in any one of the two controllers. Furthermore, this fault-tolerant model will increase system availability and reliability which are very important system attributes especially for developing countries. Fault-tolerance is an aspect that distinguishes the proposed architecture from previous systems described in the literature. In addition, this NCS is connected to the cloud using an internet gateway to provide the system with different IoT functionalities such as data analysis and greenhouse controller reprogramming or overriding. A preliminary version of this system has been briefly

presented in [23] without in depth analysis of its fault tolerance characteristics.

The rest of this paper is organized as follows. In Sect. 2, several related works are presented. In Sect. 3, the proposed system architecture is demonstrated with the simulation setup in Sect. 4. The Riverbed simulation results and analysis are presented in Sect. 5. Fault Tolerance of the proposed architecture is studied in Sect. 6. Finally, Sect. 7 concludes the whole paper.

2 Related work

In this section, some related works to the scope of this paper, are presented. The greenhouse systems proposed in the related works can be classified into monitoring systems, and monitoring and controlling systems.

2.1 Monitoring systems

Since the flowers quality is directly influenced by any minor change in the environmental conditions, Ref. [1] conducted an intensive research about the impact of the environmental change on the flowers, and proposed a monitoring system for flowers greenhouse. ZigBee technology was used to transmit the sensors reading to a coordinator, then the information is transferred over USB to a graphical interface to be displayed to the user. A similar architecture was proposed by Srbinovska et al. [2] for a vegetable greenhouse, with the focus on attaining low power and low cost design. Another approach for the greenhouse monitoring system was proposed by Lui et al. [3] where a group of wireless sensor nodes are distributed inside the greenhouse to measure different parameters and send their readings to the sink node.

Greenhouse monitoring systems can also be integrated with IoT; for example, Ref. [7] proposed a greenhouse WSN model for a monitoring solution in 2015, where each greenhouse has different sensor nodes that send the measured values to the greenhouse coordinator. The greenhouse coordinator is physically connected to the Internet gateway. Likewise is the IoT based monitoring systems presented in [12–14]. The system presented in [15] does not use the IoT technology but also incorporates machine learning for early prediction.

2.2 Monitoring and controlling systems

In [8], a hierarchal WSN for a greenhouse monitoring and controlling system is proposed. It is composed of multiple layers of routers in a tree-shaped network, where the lowest level is for the sensor nodes distributed inside the greenhouse, and the top level is for the

network coordinator that aggregates all the sensors data. The network coordinator is in charge of taking decisions inside the greenhouse. It evaluates the sensor readings it receives and accordingly sends commands to the different actuators to adjust the environmental parameters of the greenhouse. The wireless technology used in this model is ZigBee.

Dew condensation on the leaf surface has a harmful impact on the crops; therefore, Ref. [9] designed a greenhouse monitoring and controlling system dedicated to preventing this phenomenon. The sensor nodes collect different parameters that are substituted in Barenbrug formula to calculate the dew points of the leaves based on the dew points.

Similar to the monitoring systems, IoT can be part of the monitoring and controlling system, and it can have a more significant contribution because the controlling process can be triggered remotely by the user on a real-time basis [4]. The next references discuss various monitoring and controlling systems deployed inside greenhouses and integrated with IoT.

In [5], a greenhouse monitoring and controlling system composed of sensing and execution nodes distributed inside the greenhouse, was proposed. The network inside the greenhouse is connected to a coordinator and Internet gateway over ZigBee in order to deploy the IoT concept by sending the greenhouse information to the user, presenting them on a GUI, and receiving the requests to be executed inside the greenhouse.

A wireless moisture sensor network was developed in [6] where soil moisture sensors are used to observe the condition of the soil, accordingly the water valves for irrigation are turned on or off. The collected readings are transmitted to a gateway using ZigBee transceivers. The gateway is responsible for transmitting the data using WiFi or GSM modules to the central system where the collected data are displayed on any Internet enabled device.

In [10], a system design for a vegetable greenhouse was proposed. It is composed of four modules: (1) The data collection module, which is composed of different sensors, (2) The control module, which is composed of different actuators. (3) The control core module, which is responsible for three tasks: collecting the sensors data wirelessly, forwarding these data to the gateway, and evaluating the data in order to take the required actions inside the greenhouse through the actuators. (4) The power module which powers up the other 3 modules using solar energy.

In [11], a greenhouse system is proposed; this system is composed of some specific, IoT-based, subsystems that overtake the conventional farming process. These subsystems apply enhanced irrigation system, air temperature and humidity control, growing LED light, smart apiculture, and end market connection.

All the aforementioned related works targeted the design of a system from monitoring and/or controlling a single greenhouse either locally or through the Internet. In contrast, we aim to design a greenhouse that connects several greenhouses not only through the Internet but also through a hierarchical distributed NCS to boost the system reliability and provide fault tolerance when one of the greenhouse fails.

3 Proposed system architecture

In this paper, an IoT-based greenhouse monitoring and remote control architecture that is applicable to various types of crops, is proposed. The proposed architecture does not only enable the autonomous control of the greenhouse operational conditions but also allows the owner/supervisors to remotely control the greenhouse through the Internet. The owner also monitors and keeps record of the progress throughout the plantation period of the crops inside the greenhouse. As shown in Fig. 1, the proposed IoT architecture is composed of three

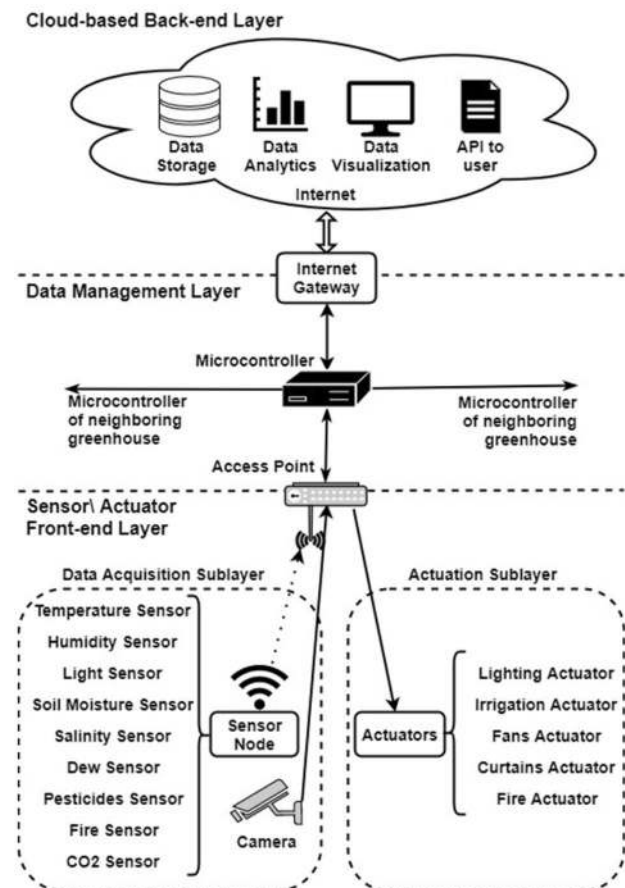


Fig. 1 Proposed system architecture

hierarchical layers: the Sensor/Actuator frontend Layer, the Data Management Layer, and Cloud-based Backend Layer. In the following subsections, all layers of the proposed architecture are discussed.

3.1 Layer 1: sensor/actuator frontend

The frontend layer of the proposed architecture is the physical interface of the system and the greenhouse environment. It is responsible for collecting the needed data and implementing the appropriate control actions. The frontend layer of the architecture is composed of two sublayers: the data acquisition sublayer, and the actuation sublayer. These two sublayers are interfaced to an access point that communicates the collected data and the needed action to the second layer of the architecture.

Data acquisition sublayer The data acquisition sublayer is responsible for gathering information from the greenhouse and passing it to the next layers of the architecture for further processing. It consists of two main components: sensor nodes, and cameras. A sensor node is a simple microcontroller that hosts a collection of sensors. Those sensors measure the different environmental parameters inside the greenhouse such as temperature, humidity, soil moisture, salinity, etc. Different sensors have different data rates according to the criticality of the measured parameter. The microcontroller of the data acquisition sublayer is equipped with a WiFi interface in order to transmit the collected data wirelessly to a local access point which relays the collected data to the next layer. The greenhouse is divided into cells. Each cell is equipped with a single access point (located in the cell's center), and sensor nodes are equally distributed to cover the surface area of the cell. In order to provide visual access to the different cells across the greenhouse, four cameras are placed in the corners of each cell to capture live video with a resolution of 5 MP and a transmission rate of 12 FPS. Each camera is connected to the access point via Ethernet cable in order to reduce the interference between its high rate traffic and the traffic of the sensor nodes.

Actuation sublayer The frontend actuation sublayer implements the control actions taken by either of the other two layers of the architecture. For instance, it changes the light conditions or the speed of the fans, and controls the irrigation valves. This sublayer is composed of a simple microcontroller interfaced with a set of actuators. In the proposed design, five actuators are used that control the lighting, irrigation, fans, and curtains inside the cell, in addition to a fire extinguisher actuator. All of the actuators take actions every 30 s except for the fire actuator which takes action every 1 s. The actuator nodes are connected to the access point of the respective cell using

Ethernet cables in order to receive the control action commands from higher layers of the architecture.

3.2 Layer 2: data management

The data management layer of the proposed architecture collects the data from the frontend data acquisition sublayer, processes such data locally and accordingly takes appropriate control actions inside the premises of the greenhouse. The control action is passed to the actuation sublayer of the frontend where it is implemented. This layer is implemented via a powerful microcontroller (referred to in this paper as the greenhouse controller) such as the Raspberry Pi, Beaglebone, or Odroid XU4 and is considered the internal brain of the greenhouse; it processes the readings gathered by the sensors, and sends commands to the actuators according to the algorithm customized for the type of the plant inside the greenhouse.

A particular owner or farming entity typically runs multiple greenhouses. In this case, each greenhouse is equipped with a single Layer 2 controller. The Layer 2 controllers of the different greenhouses share periodical watchdog signals to ensure each one's functionality. If a failure occurs in one of the greenhouse Layer 2 controllers, the data acquired from the greenhouse of the failing controller is forwarded to the controller of the nearby greenhouse. Thus, the functioning greenhouse controller takes the required actions for both greenhouses simultaneously. This increases the reliability of the greenhouse system.

3.3 Layer 3: cloud-based backend

The third layer of the proposed architecture is the cloud-based backend that allows the data of the monitored greenhouses to be accessible through the Internet. The interface between Layer 2 and this layer is implemented through an Internet gateway. Each microcontroller node at Layer 2 collects the sensor data of its greenhouse and relays such data to an Internet gateway. The Internet gateway further relays the information (after analyzing it and creating aggregating reports) to a cloud server for storage and extensive data analysis. The Internet gateway also forwards requests from the cloud server to the layer 2 microcontroller for the actuators at the physical layer nodes. Therefore, a remote greenhouse owner or supervisor, who has access to the Internet anywhere, can (1) receive a complete picture of all the information and actions taken inside the greenhouse(s) from Layer 1 and Layer 2, respectively, (2) reprogram the controller of the individual greenhouses or override the Layer 2 actions remotely over the Internet. The internet gateway is implemented using

a powerful microcontroller such as the Raspberry Pi, Beaglebone, or Odroid XU4 microcontrollers.

The backend cloud server facilitates the end-users' ability to access the sensed data and control the actions if needed. These objectives are achieved by implementing a set of services such as data storage, data analytics, data security, and data visualization. In addition, the cloud server provides an appropriate application program interface (API) and software tools through which the end-user can access the data.

4 System simulation

Using Riverbed Modeler [24], the proposed system architecture is simulated. In the following subsections, the simulation setup, the different simulation scenarios, and the performance evaluation metrics are discussed.

4.1 Simulation setup

The simulation model is composed of two greenhouses placed horizontally beside each other. Each greenhouse is 200 m in length and 40 m wide, divided into 5 square cells, 40 m × 40 m each. As shown in Fig. 2, the greenhouse's main controller is connected over Ethernet to (1) the access points of the cells (2) the Internet Gateway (3) the controllers of the neighboring greenhouses.

The greenhouse cell is a composite of sensor nodes, actuators, cameras, and an access point as shown in Figs. 3 and 4. The cameras and actuators are connected to the access point over Ethernet, while the sensor nodes are connected to the access point over WiFi IEEE-802.11n with a transmit power 5 mW. The frequency range is 5 GHz since it supports a higher number of channels than those from 2.4 GHz. Due to the abundance of channels, each cell within the same greenhouse uses a different frequency channel to eliminate interference. The channels

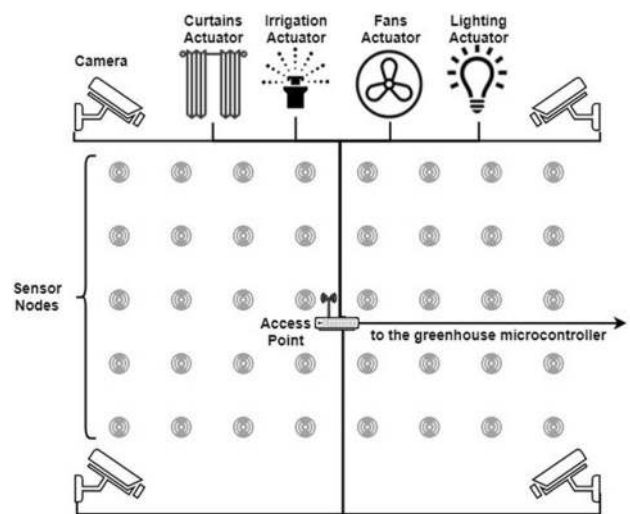


Fig. 3 Greenhouse cell

assignment of the different cells is listed in Table 1. The simulation setup parameters are summarized in Table 2.

4.2 Simulation scenarios

In order to validate the functionality and reliability of the proposed architecture, three different simulation scenarios are performed to cover all the expected situations during the operation of the greenhouse. The first scenario is the Fault-Free scenario where the main controllers of the two greenhouses are functioning properly. The other two scenarios are the Controller Failure scenarios, where the controller of one of the two greenhouses gets out of service in each scenario.

Fault-free scenario In this scenario, the sensor nodes in each of the greenhouses are sending to their respective controller only. This setup represents the daily operation of the greenhouses without any failures. In such a setup, each main controller is operational and is receiving data from all sensors and cameras in addition to distributing

Fig. 2 Greenhouse schematic

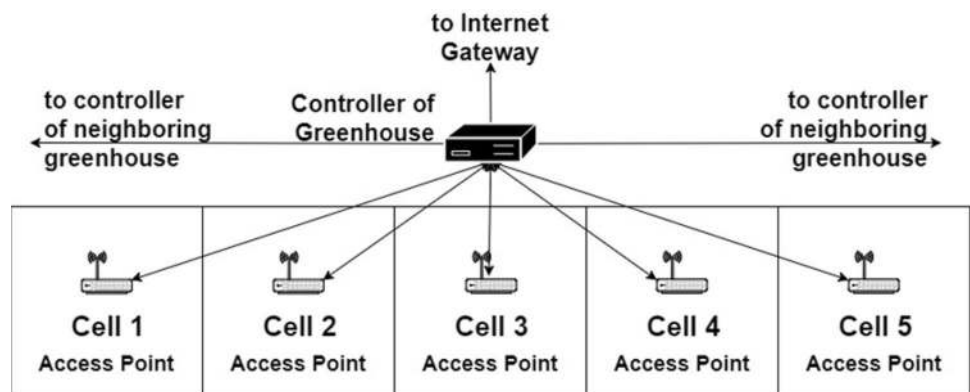


Fig. 4 Sensor nodes and access point for 1 cell

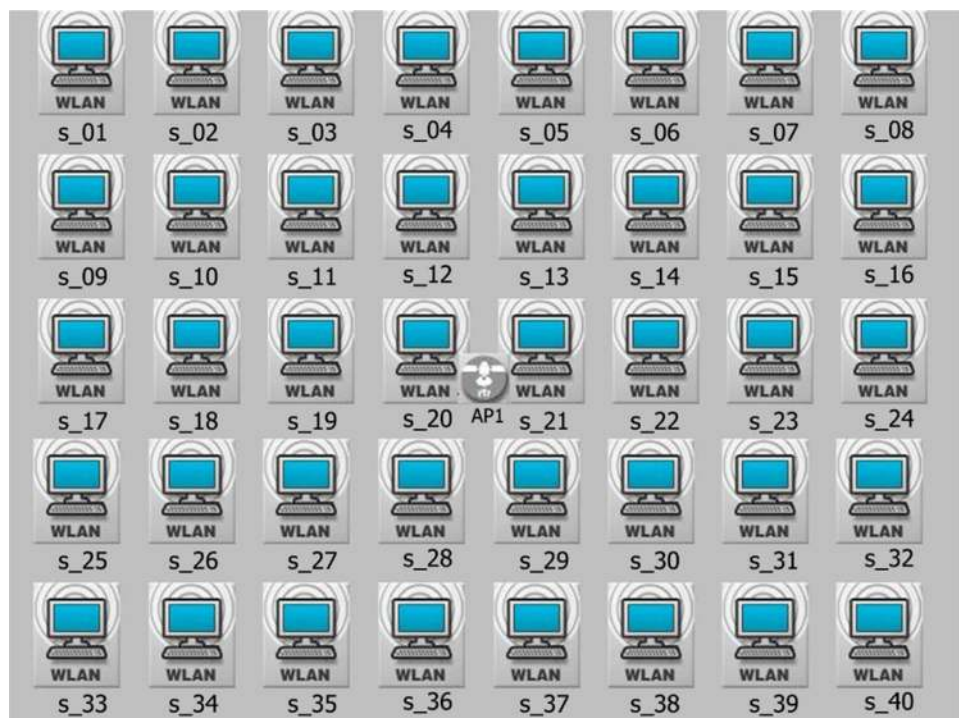


Table 1 WiFi channel assignment ID

Channel assignment	Cell 1	Cell 2	Cell 3	Cell 4	Cell 5
Greenhouse 1	56	52	48	44	40
Greenhouse 2	36	60	64	149	153

commands to the actuators in the greenhouse. Additionally, a watchdog signal is propagated between the two main controllers to acknowledge each other’s ability to operate properly. The parameters of the watchdog signal between the controllers are listed in Table 3.

Controller failure scenarios In these scenarios, the system is simulated when one of the two controllers fails. During normal operation, a watchdog signal is continuously being observed between the two greenhouses. When one of the greenhouse controllers fail, the other functioning greenhouse controller takes over the operations of both greenhouses. In this setup sensors and cameras of both greenhouses send their data to the functioning controller. Similarly, the control action to the actuators of the two greenhouses is sent from the functioning greenhouse controller.

4.3 Performance evaluation metrics

The complete information cycle inside the greenhouse starts from the sensor nodes when it sends the collected information to the main controller of the greenhouse, and ends at

the actuators that receive the required action commands from the controller. Accordingly, in order to be able to evaluate the performance of the proposed system architecture, the metrics that indicate whether the different simulation scenarios are performing as desired or not are defined.

Packet loss The first performance evaluation metric is the packet loss; packet loss in the information cycle means that some significant data will be dropped out, which will lead to either taking a wrong action due to missing in the input data, or not taking the necessary action due to losing the packets that send the commands to the actuators.

Delay Overdelayed packets in the information cycle are undesirable, because it will cause taking actions at incorrect timing; therefore the total delay for the information cycle has to be below the timeout value, and the timeout here is defined by the data rate of the fastest part of the system, which is the fire system, where the fire sensors send and the fire actuators take action every 1 s. Even though typical greenhouses tolerate relaxed sensor reading delays, a tight 1-s maximum delay is assumed due to the presence of fire sensor/actuator in the proposed system.

5 Riverbed simulation results

For each scenario, simulations were performed with 33 seeds for 1800 s, and the maximum delays and packet losses for each seed were considered and analyzed with 95% confidence.

Table 2 Simulation parameters summary

Sensor node information		
Number of sensors	9	
Sensor packet size	1 byte	
Sensor nodes transmit power	5 mW	
Sensors data rate (bytes/s)	Temperature	1/30
	Humidity	1/30
	Light	1/30
	Soil moisture	1/30
	Salinity	1/30
	Dew	1/30
	Pesticides	1/5
	Fire	1
	CO ₂	1
Actautors and cameras information		
Number of actautors	5	
Actuator packet size	10 bytes	
Actautors data rate (bytes/s)	Lighting	1/30
	Irrigation	1/30
	Fans	1/30
	Curtains	1/30
	Fire	1
	Cameras resolution	5 MP
Cameras transmission rate	12 FPS	
Cell Information		
Dimensions	40 m × 40 m	
Number sensor nodes	40	
Number of actautors	Lighting	1
	Irrigation	1
	Fans	1
	Curtains	1
	Fire	1/5
Number of cameras	4	
Wifi information		
Wireless protocol	IEEE 802.11n	
Access points transmit power	10 mW	

Table 3 Watchdog signal parameters

Parameter	Value
Packet size	1 byte
Application layer protocol	FTP
Period	1 s

5.1 Packet loss

The simulation showed that there is no packet loss at any point of time during the simulation of the three scenarios over their 33 seeds. This result confirms that the packet sizes and the data rates of the different nodes in the

proposed architecture do not make the system vulnerable to any functionality issues due to packet loss.

5.2 Delay

The measured end-to-end delay inside the greenhouse includes packet transmission, propagation, processing, and queuing delay. It is divided into different periods:

1. The first one is where the data collected by the sensor nodes are transferred from the Data Acquisition sublayer in the Sensor/Actuator Front End Layer to the main controller of the greenhouse in the Data Management Layer.
2. The second phase is where the commands taken by the main controller are transferred back to the Sensor/Actuator Front End Layer, but to the Actuation Sub-layer.
3. An additional delay is added in the case of failure of the controller of the greenhouse, since the data collected from the sensors nodes has to be forwarded to the controller of the neighboring greenhouse; similarly, the commands in the second phase are taken at the controller of the neighboring greenhouse, and returned back to their own greenhouse.

For the Fault Free scenario, it is expected to have symmetrical amounts of delay for the two greenhouses because the information takes similar paths in both greenhouses with no dependency for any of the greenhouses on the other (see Fig. 5). The periods of delay encountered in each greenhouse are (1) and (2) only.

For the controller failure scenarios, it is expected to have higher amounts of delay for the greenhouse with a failing controller due to the additional delay of forwarding data to the controller of the neighboring greenhouse, and retrieving the commands to be executed by the actuators, represented in delay period (3). Thus, the greenhouse with a failing controller encounters delay periods (1), (2), and (3), while the other greenhouse encounters (1) and (2) delays only.

The minimum and the maximum of the 95% confidence analysis for the amounts of delay, performed for the maximum delay incident per seed for the 33 seeds of each scenario are analyzed.

The results were as expected; the delays of the two greenhouses in the Fault Free scenario were comparable, while for the Controller Failure scenarios, some variation appeared. In the simulation, Controller Failure Scenario 1 has the controller of greenhouse 1 failing, and Controller Failure Scenario 2 has the controller of greenhouse 2 failing.

Fig. 5 Delay from sensor to controller in both greenhouses for one seed in error-free scenario

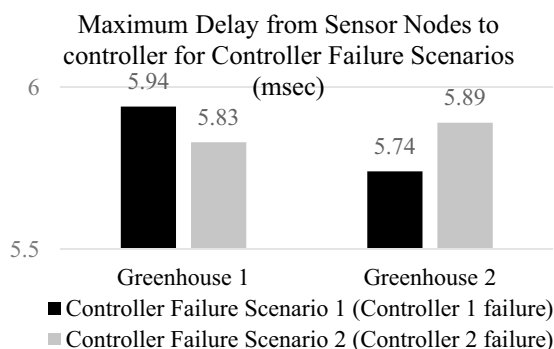
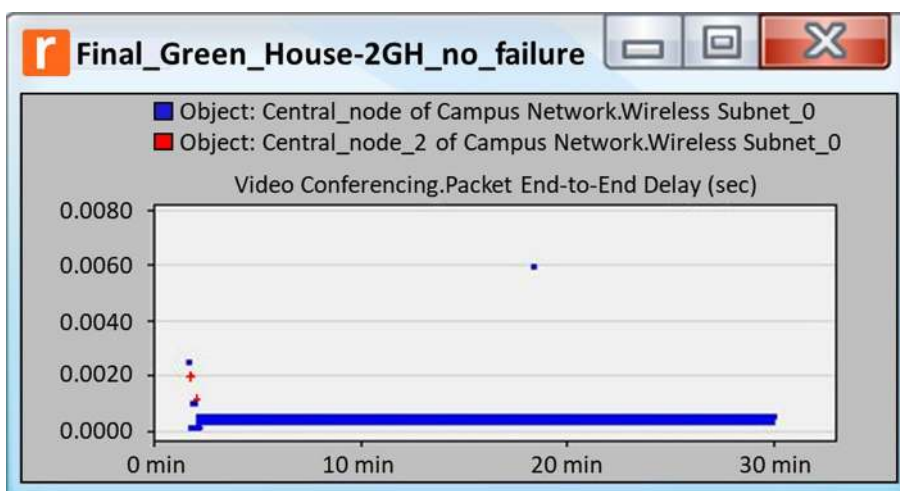


Fig. 6 Maximum delay from sensor nodes to controller for controller failure scenarios

As shown in Fig. 6, for the maximum delay from the sensor nodes to the controller, it changed for greenhouse 1 from 5.94 ms in the Controller Failure Scenario 1 to 5.83 ms in the Controller Failure Scenario 2 to with a 1.89% decrease. On the other side, it changed for greenhouse 2 from 5.74 ms in the Controller Failure Scenario 1 to 5.89 ms in the Controller Failure Scenario 2, with a 2.61% increase.

For the maximum delay from the controller to the actuators, as shown in Fig. 7, it changed for greenhouse 1 from 0.02 ms in the Controller Failure Scenario 1 to 0.01 ms in the Controller Failure Scenario 2 to with a 100% decrease. On the other side, it changed for greenhouse 2 from 0.01 ms in the Controller Failure Scenario 1 to 0.02 ms in the Controller Failure Scenario 2, with a 100% increase.

Summing up both delays in Fig. 8 for the maximum total delay, it changes for greenhouse 1 from 5.96 ms in the Controller Failure Scenario 1 to 5.84 ms in the Controller Failure Scenario 2 to with a 2.05% decrease. On the other side, it changed for greenhouse 2 from 5.75 ms in the Controller Failure Scenario 1 to 5.91 ms in the Controller Failure Scenario 2, with a 2.78% increase.

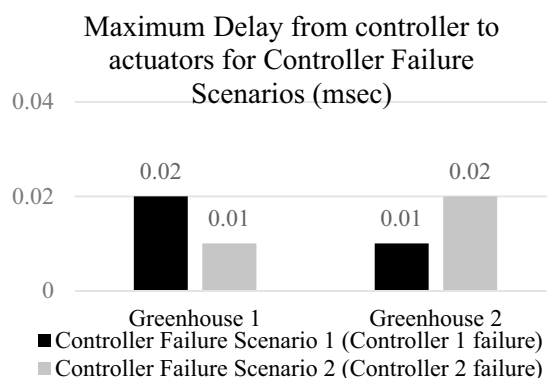


Fig. 7 Maximum delay from controller to actuators for controller failure scenarios

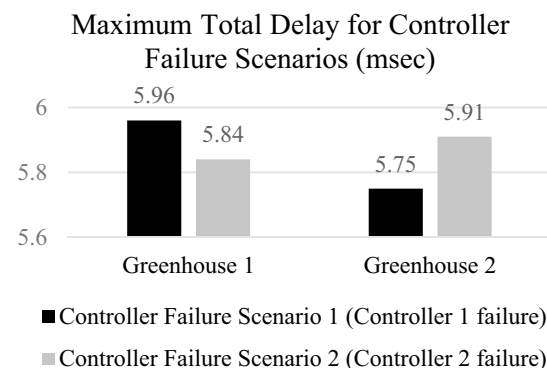


Fig. 8 Maximum total delay for controller failure scenarios

Tables 4, 5 and 6 summarize the simulation results for delay. Table 4 shows the delays for the fault free scenario, Table 5 for the controller failure scenario with the controller of the first greenhouse failing, and Table 6 for

Table 4 Fault-free scenario delay

Fault-free scenario delay	ms
<i>Greenhouse 1</i>	
Sensor nodes > controller	[5.61;6]
Controller > actuators	[0.02; 0.02]
Total delay	[5.63; 6.02]
<i>Greenhouse 2</i>	
Sensor nodes > controller	[5.54; 5.92]
Controller > actuators	[0.02; 0.02]
Total delay	[5.56; 5.94]

Table 5 Controller failure—scenario 1 delay

Controller failure scenario 1 delay (ms)	ms
<i>Greenhouse 1 (failing controller)</i>	
Sensor nodes > controller	[5.58; 5.94]
Controller > actuators	[0.02; 0.02]
Total delay	[5.6; 5.96]
<i>Greenhouse 2</i>	
Sensor nodes > controller	[5.3; 5.74]
Controller > actuators	[0.01; 0.01]
Total delay	[5.32; 5.75]

Table 6 Controller failure—scenario 2 delay

Controller failure scenario 2 delay (ms)	ms
<i>Greenhouse 1</i>	
Sensor nodes > controller	[5.44; 5.83]
Controller > actuators	[0.01; 0.01]
Total delay	[5.46; 5.84]
<i>Greenhouse 2 (failing controller)</i>	
Sensor nodes > controller	[5.52; 5.89]
Controller > actuators	[0.02; 0.02]
Total delay	[5.54; 5.91]

the controller failure scenario with the controller of the second greenhouse failing.

This variation matches the expectations of having higher amounts of delay for the greenhouse with a failing controller.

However, the maximum delay for the complete information cycle in both greenhouses for all the scenarios is negligible compared to the timeout value, which is 1 s. Accordingly, the delays for this system architecture are proven to be acceptable and not causing any functionality issues.

6 System evaluation using CTMCs

From a fault tolerance point of view, the two-greenhouse architecture described above is a 1-out-of-2 system. For the system to be operational, only one controller is required to be error-free. The system will only be subjected to a complete failure if the second controller fails before the first controller is repaired. Obviously, this fault-tolerant scheme is expected to produce much higher reliability, Mean Time to Failure (MTTF) and steady state availability when compared to those of the individual greenhouses (without any fault tolerance).

It is important to note here that controller errors may damage plants in greenhouses. This depends heavily on the type of crops grown in the greenhouse. For ornamental plants and flowers for example, variations in temperature and/or humidity can have severe effects that may lead to significant financial losses if certain actions are not taken in time. Another example of the harmful effect of controller failure can be seen when certain types of vegetables and seedlings are grown in greenhouses. In summary, a controller failure can lead to a serious downtime in the operation of a greenhouse and this can be very costly especially in developing countries. In these countries, spare parts are seldom stored on site; even dealers may often not store spare parts and only order them from abroad when needed. The time it takes to import spare parts may suffer large variations due to the unpredictability of customs inspections.

Continuous Time Markov Chains (CTMCs) will be used in this research to calculate system reliability and MTTF as well as system steady state availability. Reliability stands for the probability that the system is alive at time (t) while MTTF is the average time for the whole system to fail. Finally, the system Steady State Availability (A_{vss}) is the probability that the system is not in the failure state assuming it is repaired after a complete system failure. In these models, it is often assumed that the time to failure for a component follows the exponential distribution [25, 26]. This assumption will also be used in this work. Let λ_1 be the failure rate of the controller in K1 (the first greenhouse) and λ_2 the failure rate of K2 (the controller in the second greenhouse). In general, $\lambda_1 \neq \lambda_2$. Furthermore, let μ be the repair rate for repairing a controller. Note that $1/\mu = \text{MTTR}$, where MTTR is the Mean Time to Repair a controller.

Another important parameter in the development of the CTMC is the coverage. The coverage “ c ” is defined as the conditional probability of successful error detection and recovery when one of the two controllers (K1 or K2) fails [25–27]. Figure 9 shows this CTMC for the reliability and MTTF calculations with coverage. The model has four

states: K1K2, K1, K2 and Failure. In state K1K2, the system is working correctly with both controllers error-free. This is the initial system state. If K2 fails (with a rate $c \times \lambda_2$), the fault-tolerant system moves to state K1, i.e., K1 is the controller responsible for the operation of both greenhouses. Similar case when K1 fails, the system goes to state K2 (with a rate $c \times \lambda_1$). Accordingly, the transition from state K1K2 to state "Failure" becomes $(1 - c) \times (\lambda_1 + \lambda_2)$.

When the system in state K1 (or K2), the failure of the other controller K2 (or K1) will produce a system failure (state "Failure"). The rate of this transition is λ_1 (or λ_2) indicating the failure of K1 (or K2). In other words, if the error detection/recovery from the failure of K2 (or K1) is not successful, the entire two-greenhouse system fails and moves to state "Failure". Finally, it is important to note that the transitions from state K1 or K2 back to state K1K2 are equal to μ .

Similar to Figs. 9 and 10 shows the reliability Markov model of the two greenhouses, however, with coverage ($c = 100\%$) which is the same as the reliability model for a simple 1-out-of-2 system. Regarding the availability model of the system, Fig. 11 shows that when the system is in the Failure state, either K1 (or K2) could recover with a repair rate μ to return back to state K1 (or K2). The difference between the reliability model and the availability model is the trapping state "Failure" found in the earlier one, i.e., there is no recovery when reaching this state.

The CTMC in Fig. 11 can be represented by the following Chapman–Kolmogorov equations [25, 26]:

$$\frac{dP}{dt} = P \times T \tag{1}$$

where P is the vector of state probability functions and T is the differential state-transition rate matrix.

$$T = \begin{bmatrix} -\lambda_2 - \lambda_1 & \lambda_2 & \lambda_1 & 0 \\ \mu & -\lambda_1 - \mu & 0 & \lambda_1 \\ \mu & 0 & -\lambda_2 - \mu & \lambda_2 \\ 0 & \mu & \mu & -2\mu \end{bmatrix} \tag{2}$$

$$P = [P_{K1K2}(t) P_{K1}(t) P_{K2}(t) P_{Failure}(t)] \tag{3}$$

Knowing that $P(0) = [1 \ 0 \ 0 \ 0]$ and the summation of all probabilities is 1, the system is solved to obtain $P_{K1K2}(t)$, $P_{K1}(t)$, $P_{K2}(t)$, and $P_{Failure}(t)$. The system availability at any point of time will be equal to:

$$AV(t) = 1 - P_{Failure}(t) \tag{4}$$

Solving for the steady state availability, dP/dt in (1) is set to 0. Then the AVss is as follows:

$$AV_{ss} = 1 - \frac{\lambda_1 \lambda_2}{(\lambda_1 + \mu)(\lambda_2 + \mu)} \tag{5}$$

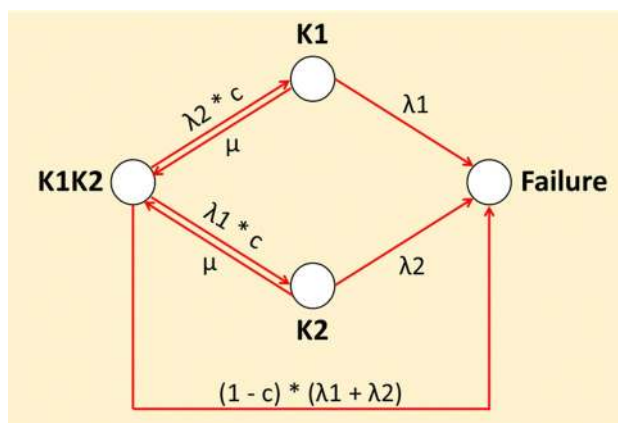


Fig. 9 Reliability Markov Model with coverage

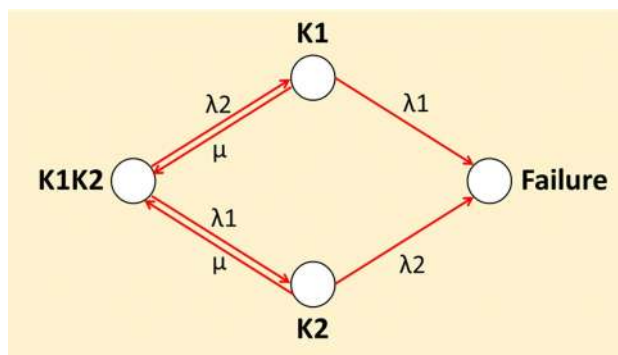


Fig. 10 Reliability Markov Model with coverage = 100%

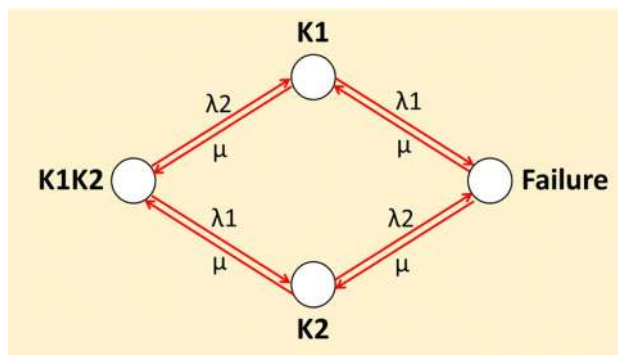


Fig. 11 Availability Markov Model

A similar analysis is used to obtain the reliability of the system which is finding the probability that the system is alive at time t . The reliability Eq. (6) is then similar to the availability Eq. (4). The same mathematical procedures starting from (1) are performed, however, with a minor

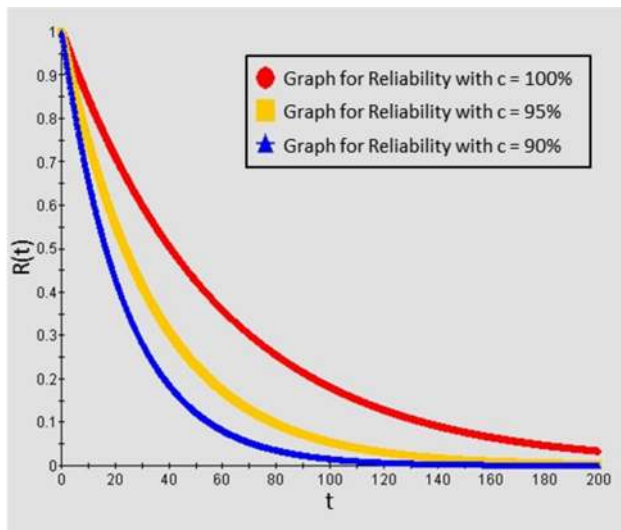


Fig. 12 Reliability versus Coverage = 100%

variation in the differential state-transition rate matrix T . The modification is changing the last row in (2) to all zeroes following the model in Fig. 10.

$$R(t) = 1 - P_{Failure}(t) \quad (6)$$

Next is a simple case study to illustrate the advantage of using this fault-tolerant scheme. Let $\lambda_1 = (1/6) \text{ month}^{-1}$ and $\lambda_2 = (1/8) \text{ month}^{-1}$. Furthermore, let $\mu = 2 \text{ month}^{-1}$. Using the software tool SHARPE [28], the reliability of the two greenhouses with respect to several values of coverage ($c = 90\%$, $c = 95\%$, $c = 100\%$) is shown in Fig. 12. The maximum reliability is obtained from 100% coverage (as expected).

Regarding the MTTF of the system with respect to changes in the coverage, Table 7 shows the coverage for 95% coverage for example is 35 months (approximately 3 years).

Finally, The SHARPE package is used to obtain the Steady State Availability (AVss) of the model in Fig. 11; it is found to be equal to 99.547%. Obviously, storing spare parts on site will increase the repair rate μ . If $\mu = 30 \text{ month}^{-1}$, AVss will increase to become 99.9977%. Without any fault tolerance in the two greenhouses, AVss for K1 and K2 would be 92.3% and 94.1% respectively if $\mu = 2 \text{ month}^{-1}$ and 99.45% and 99.59% if $\mu = 30 \text{ month}^{-1}$. Hence, storing spare parts on site will slightly increase AVss; however, if spare parts have to be imported when a failure occurs, fault tolerance has a major effect on AVss. An important conclusion to be drawn from this case study is that, if it is difficult to store spare parts on site, fault tolerance will guarantee a very high AVss.

Table 7 MTTF versus coverage

Coverage (%)	MTTF (months)
$c = 90$	24
$c = 95$	35
$c = 100$	58

7 Conclusion

Greenhouses and precision agriculture are currently a very important research topic. They are usually studied in the context of Wireless Sensor Networks (WSNs). In this paper, a relatively large two-Greenhouse system is designed. It is based on the concept of Networked Control Systems (NCSs). It is connected to the cloud. Riverbed simulations showed that the Greenhouse design was able to satisfy the required real-time constraints since some sensors had a 1 s sampling rate. The channel allocation scheme used in the design prevented interference from affecting the system. Fault tolerance was then examined. Riverbed simulations again showed that, in the event of the failure of one of the controllers in one Greenhouse, the operating controller in the other Greenhouse was able to operate the entire system correctly.

Reliability and availability were then investigated. Continuous Time Markov Chains (CTMSs) were developed to calculate both system reliability and system steady state availability. The Coverage was taken into account in all calculations. Lastly, a case study was presented and it showed the improvement in reliability and steady state availability due to the introduction of fault tolerance. The Mean Time to Failure (MTTF) was also calculated. This specific case study highlighted the fact that fault tolerance can significantly reduce the downtime due to the fact that, in many developing countries, spare parts are not stored on site and the repair time is relatively large.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Human and animal rights The research presented herein does not involve human participants and/or animals.

References

1. Erazo M, Rivas D, Perez M, Galarza O, Bautista V, Huerta M, Rojo JL (2015) Design and implementation of a wireless sensor network for rose greenhouses monitoring. In: Proceedings of the

- international conference on automation, robotics and applications, Queenstown, Feb 2015
2. Srbínovska M, Gavrovski C, Dimcević V, Krkoleva A, Borozan V (2014) Environmental parameters monitoring in precision agriculture using wireless sensor networks. *J Clean Prod* 88:297–307
 3. Lui H, Meng Z, Cui S (2007) A wireless sensor network prototype for environmental monitoring in greenhouses. In: Proceedings of the international conference on wireless communications, networking and mobile computing (WiCom), Shanghai, Sept 2007
 4. Guo T, Zhong W (2015) Design and implementation of the span greenhouse agriculture internet of things system. In: Proceedings of the international conference on fluid power and mechatronics, Harbin, Aug 2015
 5. Dan L, Xin C, Chongwei H, Liangliang JI (2015) Intelligent agriculture greenhouse environment monitoring system based on IoT technology. In: Proceedings of the international conference on intelligent transportation, big data and smart city, Halong Bay, Dec 2015
 6. Mat I, Kassim M, Harun A, Yusoff I (2016) IoT in precision agriculture applications using wireless moisture sensor network. In: Proceedings of the IEEE conference on open systems (ICOS), Langkawi, Oct 2016
 7. Gomes T, Brito J, Abreu H, Gomes H, Cabral J (2015) GreenMon: an efficient wireless sensor network monitoring solution for greenhouses. In: Proceedings of the IEEE international conference on industrial technology (ICIT), Seville, Mar 2015
 8. Sampaio H, Motoyama S (2017) Implementation of a greenhouse monitoring system using hierarchical wireless sensor network. In: Proceedings of the IEEE 9th Latin-American conference on communications (LATINCOM), Guatemala City, Nov 2017
 9. Park D-H, Park J-W (2011) Wireless sensor network-based greenhouse environment monitoring and automatic control system for dew condensation prevention. *Sensors* 11:3640–3651
 10. Bai Q, Jin C (2017) The remote monitoring system of vegetable greenhouse. In: Proceedings of the international symposium on computational intelligence and design, Hangzhou, Dec 2017
 11. Kodali RK, Jain V, Karagwal S (2016) IoT based smart greenhouse. In: Proceedings of the IEEE humanitarian technology conference (R10-HTC), Agra, Dec 2016
 12. Xu Z, Chen J, Wang Y, Fan Z (2016) A remote monitoring system for greenhouse based on the internet of things. In: Proceedings of international conference on mechanics and mechatronics research (ICMMR), Chongqing, June 2016
 13. Muthupavithran S, Akash S, Ranjithkumar P (2016) Greenhouse monitoring using internet of things. *Int J Innov Res Comput Sci Eng (IJIRCSE)* 2(3):13–19
 14. Akkaş MA, Sokullu R (2017) An IoT-based greenhouse monitoring system with Micaz motes. *Procedia Computer Science, Elsevier* 113:603–608
 15. Hossam M, Kamal M, Moawad M, Maher M, Salah M, Abady Y, Hesham A, Khattab A (2018) PLANTAE: an IoT-based predictive platform for precision agriculture. In: Proceedings of IEEE Japan–Africa conference on electronics, communications and computers (JAC-ECC), Alexandria, Dec 2018
 16. Skeie T, Johannessen S, Brunner C (2002) Ethernet in substation automation. *IEEE Control Syst* 22(3):43–51
 17. Decotignie JD (2005) Ethernet-based real-time and industrial communications. *Proc IEEE* 93(6):1102–1117
 18. Felsler M (2005) Real-time ethernet-industry prospective. *Proc IEEE* 93(6):1118–1129
 19. Steigmann R, Endresen J (2006) Introduction to WISA: WISA-Wireless interface for sensors and actuators. White paper, ABB, Calgary
 20. Lian FL, Moyne JR, Tilbury DM (2001) Networked control systems toolkit: a simulation package for analysis and design of control systems with network communication. In: Tech Rep, UM-ME-01-04, July 2001
 21. Nilsson J (1998) Real-time control systems with delays. PhD thesis, Department of Automatic Control, Lund Institute of Technology, Lund
 22. Zhao Y-B, Sun X-M, Zhang J, Shi P (2015) Networked control systems: the communication basics and control methodologies. *Math Probl Eng* 2015:639793
 23. Ibrahim H, Mostafa N, Halawa H, Elsalamouny M, Daoud R, Amer H, Shaarawi A, Khattab A, ElSayed H (2018) A high availability networked control system architecture for precision agriculture. In: Proceedings of IEEE international conference on computer and applications (ICCA), Beirut, July 2018
 24. Official website for Riverbed modeler [online]. Available: <https://www.riverbed.com/>. Accessed 8 Feb 2019
 25. Siewiorek DP, Swarz RS (1998) Reliable computer systems design and evaluation. A K Peters, Natick
 26. Trivedi KS, Bobbio A (2017) Reliability and availability engineering-modeling, analysis and applications. Cambridge University Press, Cambridge
 27. Amer HH, McCluskey EJ (1987) Calculation of coverage parameter. *IEEE Trans Reliab* 36:194–198
 28. Official site for SHARPE: <http://sharpe.pratt.duke.edu>. Accessed 8 Feb 2019

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.