Journal of Digital Convergence
Vol. 18. No. 3, pp. 171-177, 2020

ISSN 2713-6434 / eISSN 2713-6442
https://doi.org/10.14400/JDC.2020.18.3.171

효율적인 클라우드 기반 스마트팜 제어 시스템 구현 방법

최민석
삼육대학교 경영정보학과 교수

A Study on the Efficient Implementation Method of Cloud-based Smart Farm Control System

Minseok Choi
Professor, Department of Management Information Systems, Sahmyook University

요 약 4차 산업혁명의 영향으로 농축산업 분야에도 정보통신 기술을 융합한 스마트팜 기술의 도입을 통한 생산성 증대 및 경쟁력 강화를 추진하고 있다. 이러한 스마트 농업 기술은 농업 분야의 미래 성장을 위한 새로운 패러다임으로 자리 잡고 있다. 스마트팜 구현을 위하여 실시간 재배 환경 모니터링과 이와 연계된 재배 환경 자동 제어 시스템의 개발이 필요하며 나아가 작물 생육 상태를 확인하여 관리하는 지능형 시스템의 개발이 요구되어 진다. 본 논문에서는 호환성과 확장성이 뛰어난 웹 플랫폼을 활용하여 클라우드 기반 스마트팜 관리 시스템 구현을 위한 빠르고 효율적인 개발 방법을 제안한다. 제안된 웹 플랫폼을 이용한 개발 방법이 효율적이며 안정적인 시스템 구현이 가능함을 실제 구현된 시스템의 운영을 통하여 확인하였다.

주제어 : 스마트 팜, 클라우드 팜, 지능형 작물 재배, 시설재배, 농업

Abstract Under the influence of the Fourth Industrial Revolution, there are many tries to promote productivity enhancement and competitiveness by adapting smart farm technology that converges ICT technologies in agriculture. This smart farming technology is emerging as a new paradigm for future growth in agriculture. The development of real-time cultivation environment monitoring and automatic control system is needed to implement smart farm. Furthermore, the development of intelligent system that manages cultivation environment using monitoring data of the growth of crops is required. In this paper, a fast and efficient development method for implementing a cloud-based smart farm management system using a highly compatible and scalable web platform is proposed. It was verified that the proposed method using the web platform is effective and stable system implementation through the operation of the actual implementation system.

Key Words : Smart Farm, Cloud Farm, Intelligent Cultivation, Greenhouse, Agriculture

1. 서론

정보통신기술(ICT)의 발전, 특히 IOT(Internet of Things) 기반 기술의 발전은 우리의 생활 곳곳에 스며들어 삶의 질을 향상시키고 있으며, 다양한 분야로 활용 범위가 확대되고 있다. 1차 산업인 농업 분야에서도 ICT 기술을 활용한 스마트팜을 통하여 인구 감소와 고령화에 따른 여러 문제해결 및 농업 경쟁력 강화를 모색하고 있다[1,2]. 스마트농업은 ICT 기술 융합을 통한 농업 분야의 혁신과 지속적 성장을 이끌기 위한 미래 농업의 새로운 패러다임으로 주목받고 있다[3].

농부의 경험과 주관적 판단에 의존하던 농업 기술을

*Corresponding Author : Minseok Choi(mschoi@syu.ac.kr)
Received December 26, 2019 Revised January 22, 2020
Accepted March 20, 2020 Published March 28, 2020

정량화된 데이터와 인공지능에 의한 자동화 시스템으로 대체하며 네트워크 연결을 통하여 시간과 장소의 구애 없이 관리할 수 있게 하는 스마트팜을 미래 농업의 모델로 인식하고 이를 성장산업으로 키우기 위하여 세계 각국이 전략적으로 기술개발을 위하여 나서고 있으며 정부에서도 패러다임 전환의 필요성을 인지하고 ICT 기술 융합을 통한 한국형 스마트팜 개발 및 보급을 위한 노력을 지속하고 있다[4]. 농업 분야와 ICT의 융복합 기술개발 및 보급에 대한 정책은 2004년 정보통신부에서 시작되어 2010년부터 농림수산식품부 주도로 추진되고 있으며, 최근 정부는 4차 산업혁명 기반의 8대 선도사업에 농업 분야의 스마트팜 사업을 포함했다[5].

현재까지의 적용사례를 보면 일부 시설원예 분야를 제외하면 실시간 환경 모니터링과 다양한 제어기능이 연계된 스마트팜 시스템의 개발 및 적용은 다소 부족한 편이며, 농작물의 생육 상태 및 환경 모니터링과 자동 제어를 기반으로 하여 작물 생산의 전체 주기를 지능적으로 관리해주는 스마트팜 시스템의 개발 및 보급이 요구된다.[6]. 또한, 국외의 경우 이미 상용 스마트팜 시스템들이 시장을 확장하고 있으므로 단기간에 관련 기술의 경쟁력 확보가 필요하다.

본 논문에서는 높은 호환성과 확장성을 가지는 웹 플랫폼을 활용하여 클라우드 기반 스마트팜 제어 시스템 구현을 위한 빠르고 효율적인 개발 방법을 제안하고자 한다.

2. 스마트팜 개요 및 현황

한국농촌경제연구원에서는 스마트팜을 ICT 기술을 농축산 산업에 접목하여 원격에서 자동으로 작물과 가축의 환경을 유지 및 관리할 수 있는 농장이라고 규정하였는데, 이는 작물의 생육 및 환경에 대한 데이터를 기반으로 시간과 장소에 구애받지 않고 작물이나 가축의 생육환경을 확인하고 조정함으로써 농축산물의 생산성과 품질 제고가 가능한 농업을 말한다[4,7].

스마트팜은 Fig. 1과 같이 온도, 습도, 일사량, CO2 등의 환경데이터를 수집하는 환경정보 모니터링 기능과 냉방, 난방, 환기, CO2, 양분, 사료 공급 등의 자동 및 원격 환경 제어 기능 그리고 이 두 가지를 연동하는 복합환경 제어 시스템으로 구성되며, 이중 생산성 제고 및 이익증대와 직결되는 복합 환경 제어 시스템의 구현이 핵심 요소이다[8].

Fig. 1. Structure of environment control system for smart farm [9]

기존 연구에서 스마트팜 제어 시스템 구현을 위하여 범용 임베디드 컴퓨팅 플랫폼을 이용한 스마트팜 설계 방법이 제안되었지만[10], 하드웨어 설계에 초점이 맞추어졌으며, 임베디드 플랫폼에서 네이티브 소프트웨어 개발과정은 복잡하고 난도가 높아 개발 효율이 떨어진다. 상용 클라우드 기반 스마트팜 제어 시스템으로 인지도가 높은 캐나다 LINK4 Greenhouse Controls의 제품군도 Fig. 2와 같이 전용 하드웨어와 네이티브 소프트웨어로 구현되어 확장성이 낮은 편이다[11].

Fig. 2. Link4's Hydroponic Controller [11]

3. 클라우드 기반 스마트팜 제어 시스템 구현

3.1 시스템 구성

본 논문에서 제안하는 스파트팜 제어 시스템은 Fig. 3과 같이 각 시설에서 환경정보 모니터링 및 제어를 담당할 로컬 관리 시스템(Farm Client)과 인터넷을 통하여 연결된 로컬 관리 시스템들을 통합 관리하고 환경데이터를 저장하는 클라우드 관리 시스템(Cloud Farm Server) 그리고 PC 및 모바일 환경의 사용자 애플리케이션으로 구성된다.

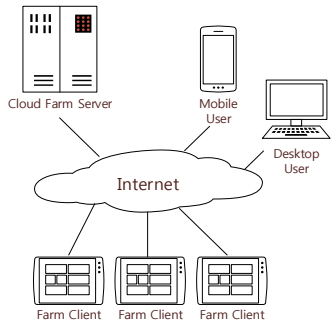

Fig. 3. Configuration of proposed smart farm system

3.2 로컬 관리 시스템 H/W 구현

로컬 관리 시스템은 단위 스마트팜 시설에서 데이터 수집, 모니터링 및 환경 제어를 담당하는 시스템으로 Fig. 4와 같이 하드웨어를 구성하였다.

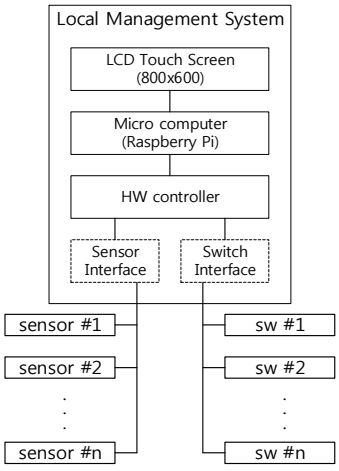

Fig. 4. H/W structure of local management system

하드웨어 플랫폼은 마이크로컴퓨터로 Raspberry Pi 3 Model B+[12]를 사용하였고, 사용자 인터페이스 구현을 위한 입출력 장치로는 800x600 LCD 터치스크린을 연결하였다. 안정적 하드웨어 운영을 위하여 각종 센서 입력과 장치 조작을 위한 릴레이 스위치 제어를 담당할 컨트롤러 보드를 별도로 구성하고 Raspberry Pi와 시리얼 인터페이스로 연결하였다. 하드웨어 컨트롤러를 독립적으로 구성하여 마이크로컴퓨터에 문제가 생겼을 때 연결된 개별 장비의 제어 설정에 영향을 주지 않도록 하였으며, 비상시 각 스위치를 수동조작이 가능하도록 하였다. Fig. 5는 실제 구현된 로컬 관리 시스템의 하드웨어 프로토타입의 모습을 보여준다. 그림의 위쪽은 LCD 터치스크린과 연결된 Raspberry Pi이며 아래는 센서 연결을 위한 커넥터와 릴레이 스위치들을 가진 하드웨어 컨트롤 보드이다.

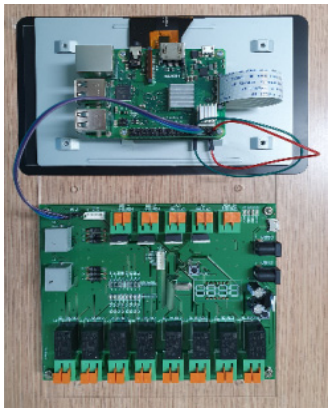

Fig. 5. Prototype of local management system

3.3 로컬 관리 시스템 S/W 구현

로컬 관리 시스템의 운영 프로그램 구축을 위한 소프트웨어 구성은 Fig. 6과 같다. Raspberry Pi의 운영체제로는 리눅스 기반의 Raspbian[13]을 설치하였고, 각종 설정과 환경데이터 저장에 사용될 데이터베이스 구축을 위한 데이터베이스 관리 시스템으로 MySQL을 사용하였다.

웹 기반 사용자 인터페이스(UI) 구현을 위하여 로컬 웹 서버로 NGINX[14]를 사용하였고, 웹 브라우저로 Chromium[15]을 키오스크(kiosk) 모드로 실행하여 LCD 스크린에 사용자 인터페이스를 표현하였다. UI 구현은 PHP와 HTML5, JavaScript를 이용하여 구현하였다. 웹 기반 사용자 인터페이스 구현의 장점은 네이티브 애플리케이션의 구현에 비하여 개발 작업의 난도는 낮지만 높은 품질의 UI 구현이 가능하며 로컬 관리 시스템 외에도 클라우드 기반의 사용자 애플리케이션의 구현도 같은 방식으로 가능하여 개발 및 유지보수 작업의 효율성을 높일 수 있다.

모니터링 및 환경 자동 제어 기능의 구현은 하나의 통합 프로그램으로 구현하지 않고 단위 작업을 구현 후 리눅스의 Cron을 이용하여 주기적으로 각 단위 작업을 순차 실행하도록 구현하였다. 이러한 방식은 프로그램의 구조를 단순화하고 메모리 사용 및 관리를 효율화하여 시스템과 프로그램의 안정성을 높일 수 있다.

로컬 관리 시스템의 각종 설정 관리 및 각 단위 작업 사이의 데이터 공유는 MySQL을 이용하여 로컬 데이터베이스를 구성하여 처리하였다. 로컬 데이터베이스를 이용함으로써 서버와의 통신 장애가 발생하여도 각종 환경 데이터를 일정 기간 손실 없이 유지할 수 있으며, 독자적으로 관리 시스템의 운영이 가능하게 된다.

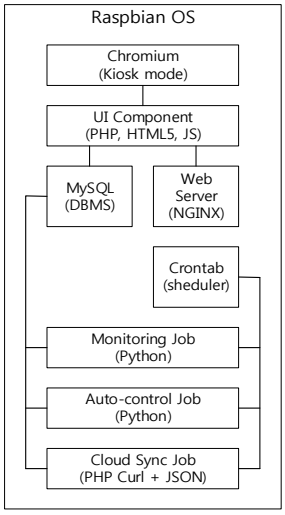

Fig. 6. S/W architecture of local management system

3.4 클라우드 관리 시스템 구현

로컬 관리 시스템들을 통합 관리하는 클라우드 관리 시스템은 환경데이터 저장을 위한 데이터베이스 서버와 웹 기반의 통합 관리 서버로 구성되어 진다. 기본 서버 구성은 아마존의 클라우드 컴퓨팅 솔루션인 AWS(Amazon Web Services)[16]를 기반으로 하였다. Fig. 7과 같이 데이터베이스 서버는 클라우드 기반의 MySQL 호환 관계형 데이터베이스 서비스인 Amazon Aurora[17]를 이용하였고, 웹 기반 관리 서버는 Amazon EC2[18] 서비스를 이용하여 다중 서버를 구성하였다.

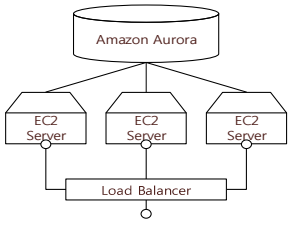

Fig. 7. Configuration of Cloud Server on AWS

클라우드 관리 시스템의 데이터베이스는 로컬 관리 시스템의 데이터베이스 구조와 같게 설계하였으며, 각 클라이언트 시스템을 구분하기 위한 고유 단말 번호만을 추가하였다. 로컬 시스템에서 클라우드 서버로 동기화를 위한 데이터 전송은 HTTP 프로토콜 기반의 REST 구조를 사용하였다. 각 단말은 실시간 환경데이터, 환경 설정, 장비 조작 등의 정보를 JSON 형식으로 가공하여 클라우드 웹 서버로 전송하며, 웹 서버는 전송받은 데이터를 데이터베이스에 동기화한다.

사용자가 원격에서 클라우드 서버에 접속하여 로컬 시스템을 조작할 경우 클라우드 서버의 설정이 로컬 관리 시스템으로 동기화되어야 하는데, 이 경우에는 앞의 경우와 반대로 서버에서 클라이언트의 로컬 웹서버로 REST 방식으로 동기화 데이터를 전송하고 로컬 웹서버가 전송받은 데이터를 시스템에 동기화한다.

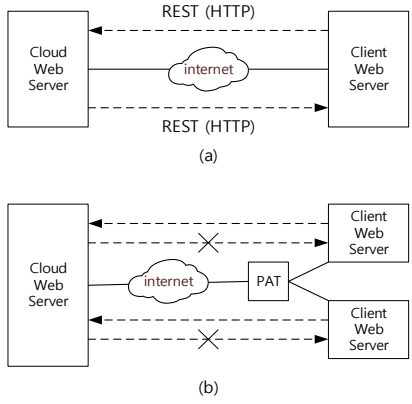

Fig. 8. Architecture of Synchronization between server and client

Fig. 8은 서버와 클라이언트 사이의 동기화 구조를 간략히 보여주고 있다. 이때 Fig. 8 (b)와 같이 클라이언트가 유무선 공유기 같은 NAT(network address translation)이나 PAT(port address translation) 장비에 연결되어 있을 때는 HTTP 프로토콜의 특성상 서버에서 클라이언트로 연결할 수 없게 되어 문제가 발생하게 된다.

대부분 로컬 관리 시스템은 독립적으로 IP를 부여받아 접속하지 않고 IP 공유기 등을 거쳐 인터넷에 접속하게 된다. 따라서 이러한 문제를 해결하지 못하면 사용자의 원격 접속에 의한 실시간 동기화가 어렵게 된다.

서버에서 클라이언트로의 동기화 문제를 해결하기 위하여 Fig. 9와 같이 클라이언트에서 서버로 소켓 터널링을 구현하였다. 클라이언트에서 서버로 연결형 소켓 터널

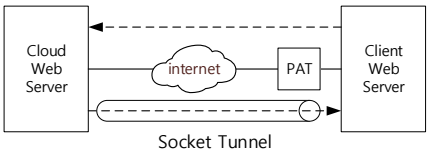

Fig. 9. Socket Tunneling

을 생성하면 PAT 장비가 존재하여도 터널을 통하여 서버에서 클라이언트로의 HTTP를 통한 REST 요청이 가능해진다. 이를 이용하여 사용자가 클라우드 서버를 통하여 원격에서 조작 명령을 실행할 경우 바로 로컬 단말로 명령이 동기화되어 실행될 수 있다.

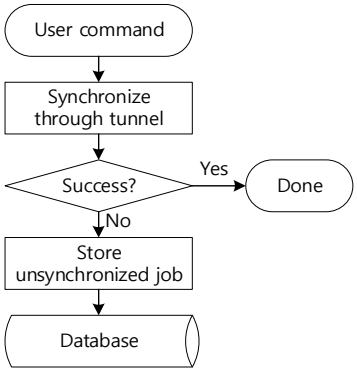

Fig. 10. Flowchart of server to client synchronization process

만약 터널링에 오류가 발생하면 서버에서 클라이언트로의 동기화가 되지 않는 문제가 여전히 존재하게 된다. 이를 예방하기 위하여 사용자 조작에 의한 서버에서 클라이언트로의 동기화 처리 과정을 Fig. 10과 같이 이중화하여 터널을 통한 동기화가 실패하였을 경우 동기화되지 않은 작업을 DB에 저장한 후, Fig. 11과 같이 클라이언트에서 동기화 요청을 받으면 이를 처리한 후 DB에 동기화되지 않은 작업이 존재할 경우 이를 요청에 대한 응답으로 클라이언트에 전송하여 동기화를 진행하도록 한다. 클라이언트의 경우 응답을 확인하여 서버로부터 동기화되지 않은 작업이 있을 때 이를 동기화하고 소켓 터널을 다시 설정하여 이후 실시간 동기화가 가능하도록 한다.

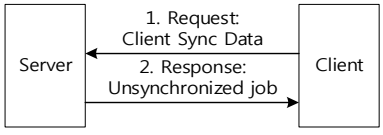

Fig. 11. Client to server synchronization cycle (REST)

3.5 통합 UI 구현

3.3절의 로컬 관리 시스템의 사용자 프로그램 UI 구현에서 언급하였듯이 클라우드 서버에서 사용자들에게 제공하는 프로그램 UI는 로컬 관리 시스템의 UI 구현 소스를 일부 수정 후 사용할 수 있다. 로컬 관리 시스템의 프로그램 UI를 웹 기반으로 개발하였고 상태 정보를 DB에서 불러와 표시하는 방식을 사용하고 있으므로, 클라이언트와 같은 DB 구조의 클라우드 서버에서도 각 단말을 식별하기 위한 부분만 추가하면 그대로 사용하는 것이 가능하다. HTML5를 기반으로 정보와 뷰(view)를 분리하도록 구현하였기 때문에 CSS(cascade style sheet)를 변경하여 PC 화면에 맞는 뷰를 구현하였으며, 반응형 디자인(responsive design)을 적용하여 스마트폰 등의 모바일 뷰도 구현하였다. Fig. 12는 로컬 관리 시스템 단말의 LCD에 구현된 사용자 프로그램 UI 모습이며, Fig. 13은 클라우드 서버에 PC 접속 시 프로그램 UI, Fig. 14는 모바일 접속 시의 프로그램 UI를 보여준다.

Fig. 12. UI of local management system

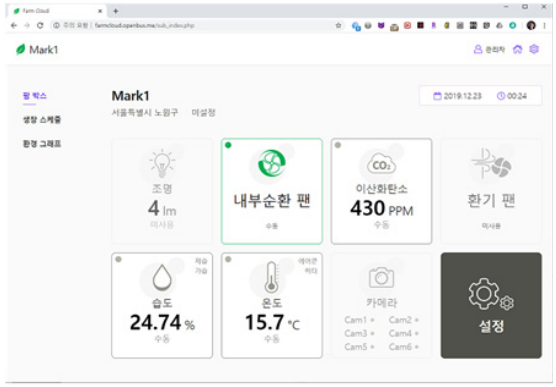

Fig. 13. UI of cloud service for PC

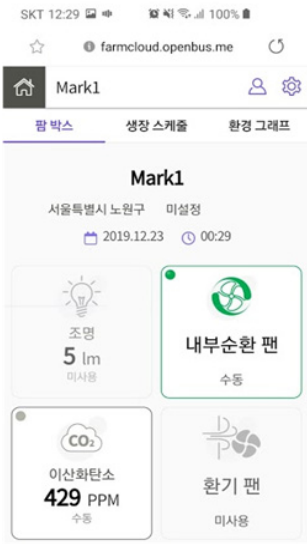

Fig. 14. UI of cloud service for mobile

4. 고찰 및 결론

4차 산업혁명의 물결은 1차 산업인 농축산업에도 패러다임의 변화를 일으키고 있다. ICT 기술과 융합된 스마트팜의 도입은 침체된 농업환경의 개선에 크게 이바지할 것으로 기대되고 있으며, 관련 기술의 확보를 위하여 정부 차원에서도 많은 지원과 노력을 하고 있다. 특히 스마트팜 기술의 고도화 및 지능화를 통하여 농작물의 생육 상태 및 환경 모니터링과 자동 제어를 기반으로 하여 작물 생산의 전체 주기를 지능적으로 관리해주는 시스템의 개발 및 보급이 요구되지만, 국내 관련 기술은 미흡한 편이다.

본 논문에서는 클라우드 기반의 스마트팜 제어 시스템을 개발하기 위한 효율적인 방법을 제안하였다. 범용 마이크로컴퓨터인 Raspberry Pi와 리눅스 기반의 Raspbian OS를 사용하여 로컬 단말의 제어 시스템을 설계하였고, 웹 기반으로 단말의 운영 프로그램을 구현하여 개발 효율 및 안정성을 높였다. 클라우드 서버도 로컬 단말을 가상화하여 확장하는 방식으로 설계하여 단말과 서버의 운영 프로그램 코드가 공유될 수 있도록 하여 개발 및 유지보수 비용을 낮출 수 있도록 하였다. 서버와 클라이언트 사이의 동기화를 위한 통신은 검증된 HTTP 프로토콜 기반의 REST 방식을 이용하며, 공유기 사용에 따른 서버에서 클라이언트로의 접속 문제를 해결하기 위하여 소켓 터널링을 이용하였다. 클라이언트에 로컬 데이터베이스를 구성하여 네트워크 접속 장애가 발생하여도 독자적 운영 및 데이터 수집이 가능하도록 하였고, 서버와 클라이언트 간의 동기화 방법을 이중화하여 실시간 동기화에 문제가 발생하여도 DB를 이용하여 추후 동기화가 가능하도록 하였다.

구현된 시스템의 검증을 위하여 5곳의 재배 시설에 적용하여 4개월 이상을 운영하며 광량, 온도, 습도, CO2 농도로 구성된 환경 센서 측정 데이터 세트를 380만 건 이상 누적시켰으며, 이를 통하여 안정적 운영을 확인하였다.

앞으로의 과제는 작물의 생육 상태 데이터를 자동으로 측정 및 수집하는 기술의 개발이 요구되며 나아가 누적된 환경 및 생육 데이터를 활용하여 최적의 재배 환경을 자동으로 제어하는 지능형 제어 시스템으로의 확장이 필요할 것으로 생각된다.

REFERENCES

[1] H. S. Kim, D. D. Lee & H. S. Kim. (2014). *Strategies and Tasks of ICT Convergence for the Creative Agriculture Realization(R736)*, Seoul: Korea Rural Economic Institute.

[2] T. Y. Lee & C. M. Heo. (2019). A Study on the Influence of Acceptance Factors of ICT Convergence Technology on the Intention of Acceptance in Agriculture : Focusing on the Moderating Effect of Innovation Resistance. *Journal of Digital Convergence, 17(9)*, 115-126.

[3] M. H. Ahn & C. M. Heo. (2019). The Effect of Technical Characteristics of Smart Farm on Acceptance Intention by Mediating Effect of Effort Expectation. *Journal of Digital Convergence, 17(6)*, 145-157.

[4] N. G. Yoon, J. S. Lee, G. S. Park & J. Y. Lee. (2017). Korea smart farm policy and technology development status. *Rural Resources, 59(2)*, 19-27.

[5] Ministry of Economy and Finance. (2018. 08. 13). *5th Innovation Growth Ministerial Meeting-Strategic Investment Direction for Innovation Growth*. MOEF(Online), http://www.moef.go.kr/nw/nes/detailNesDtaView.do?menuNo=4010100&searchNttId1=MOSF_000000000018581&searchBbsId1=MOSFBBS_000000000028

[6] U. H. Yeo, I. B. Lee, K. S. Kwon, T. H Ha, S. J. Park, R. W. Kim, and S. Y. Lee. (2016). Analysis of Research Trend and Core Technologies Based on ICT to Materialize Smart-farm. *Protected Horticulture and Plant Factory, 25(1)*, 30-41.

[7] J. Y. Yoon & B. H. Lee. (2017). Implementation strategy and development methods for smart farms in Gangwon Province. *Journal of Agricultural, Life and*

Environmental Sciences, 29, 137-151.

[8] S. J. Oh. (2017). A Design of intelligent information system for greenhouse cultivation. *Journal of Digital Convergence, 15(2)*, 183-190.

[9] Smart Farm Korea. (2019. 10). *Structure of smart greenhouse*. EPIS(online). https://www.smartfarmkorea.net/contents/view.do?menuId=M01010103

[10] S. H. Lee. (2018). The Fundamental Functionality Design of a Smart Farm Using an Embedded Computing Platform. *Journal of The Institute of Electronics and Information Engineers, 55(4)*, 557-563.

[11] LINK4. (2019. 10). *LINK4 Cloud-Based Greenhouse Controls*. LINK4(online). http://link4controls.com

[12] RASPBERRY PI FOUNDATION. (2019. 10). *Raspberry Pi 3 Model B+*. RASPBERRY PI FOUNDATION(online). https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/

[13] RASPBERRY PI FOUNDATION. (2019. 10). *Raspbian*, RASPBERRY PI FOUNDATION. https://www.raspberrypi.org/downloads/raspbian/

[14] NGiNX. (2019. 10). *Learn how to configure caching, load balancing, cloud deployments, and other critical NGINX features*. NGiNX(online). https://nginx.org/en/

[15] The Chromium Projects. (2019. 10). *chromium*. The Chromium Projects(online). https://www.chromium.org/Home

[16] Amazon. (2019. 10). *Amazon Web Services*. AWS(online). https://aws.amazon.com/ko/

[17] Amazon. (2019. 10). *Amazon Aurora*. AWS(online). https://aws.amazon.com/ko/rds/aurora/?hp=tile&so-exp=below

[18] Amazon. (2019. 10). *Amazon EC2*. AWS(online). https://aws.amazon.com/ko/ec2/?hp=tile&so-exp=below

최 민 석(Minseok Choi) [정회원]

· 1996년 2월 : 한양대학교 전자공학과 (공학사)
· 1998년 8월 : 한양대학교 전자공학과 (공학석사)
· 2004년 8월 : 한양대학교 전자공학과 (공학박사)
· 2012년 3월 ~ 현재 : 삼육대학교 경영정보학과 교수
· 관심분야 : 정보처리, 정보시스템
· E-Mail : mschoi@syu.ac.kr