International Journal of Internet, Broadcasting and Communication Vol.12 No.4 109-121 (2020)
http://dx.doi.org/10.7236/IJIBC.2020.12.4.109

IJIBC 20-4-14

Design of Remote Management System for Smart Factory

Heejoung Hwang

Professor, Dept. of Computer Engineering, Gachon University, KOREA
hwanghj@gachon.ac.kr

Abstract

As a decrease in labor became a serious issue in the manufacturing industry, smart factory technology, which combines IT and the manufacturing business, began to attract attention as a solution. In this study, we have designed and implemented a real-time remote management system for smart factories, which is connected to an IoT sensor and gateway, for plastic manufacturing plants. By implementing the REST API in which an IoT sensor and smart gateway can communicate, the system enabled the data measured from the IoT sensor and equipment status data to the real-time monitoring system through the gateway. Also, a web-based management dashboard enabled remote monitoring and control of the equipment and raw material processing status. A comparative analysis experiment was conducted on the suggested system for the difference in processing speed based on equipment and measurement data number change. The experiment confirmed that saving equipment measurement data using cache mechanisim offered faster processing speed. Through the result our works can provide the basic framework to factory which need implement remote management system.

Keywords: *Smart Factory, Remote Management, API, REST, Web, Redis*

1. Introduction

As a result of the development of information technology (IT) and acceleration of the "Fourth Industrial Revolution", IT convergence technologies are drawing attention in various fields [1-3]. In particular, technology that combines IT and manufacturing is receiving interest in the manufacturing industry due to the increase in productivity, and it is developing into smart factories through the combination with internet of things (IoT) technology. Smart factory refers to a futuristic factory that manufactures customer-specific products with minimum cost and time through the convergence with information and communication technologies throughout the entire process from product planning and design to manufacturing, distribution and sales, and in this system, equipment connected to the Internet obtains and utilizes information for a more efficient manufacturing process [4]. Smart factory can reduce wasted manpower, control variables that may arise from humans, and improve efficiency and productivity [5]. However, in order to apply smart factory manufacturing, existing factories in the manufacturing business need to furnish advanced equipment, which requires partial suspension of the manufacturing process. This makes it difficult to create profit and replacing existing equipment and reestablishing infrastructure are time-consuming and very costly [6]. Therefore, ways

Manuscript Received: August. 22, 2020 / Revised: August. 27, 2020 / Accepted: September. 2, 2020
Corresponding Author: hwanghj@gachon.ac.kr
Tel: +82-31-750-4758, Fax: +82-31-750-4758
Professor, Dept. of Computer Engineering, Gachon University, KOREA

to utilize existing equipment and establish a smart factory at low cost have become necessary.

In the case of a plastic manufacturing factory, the manufacturing process consists of raw material dehumidification, raw material drying and injection molding. If the moisture ratio in plastic exceeds the standard in this process, humidity rises and condensation forms, resulting in defective products [7]. In the past, people checked numerous dehumidifying dryers to confirm that the moisture ratio of raw material didn't exceed the standard and that machines functioned properly, and this caused a waste of labor. Also, delays in checking the status of raw materials resulted in defective products, causing a decline in the production rate. To solve such problems, monitoring software technology that can control multiple dehumidifying dryers remotely and check raw ingredient status became necessary. Also, dehumidifying dryers are large and there are up to 100 dryers in a single plant, so it is very time-consuming and costly to replace them with advanced equipment.

This study designed and implemented a smart monitoring system for a factory that can communicate with IoT modules and remotely control equipment to respond to errors immediately in the manufacturing environment, where it is difficult to secure advanced technology. An experiment in a plastic manufacturing factory was conducted where the status of dehumidifying dryers measured by IoT sensors on dryers and raw ingredient data were transmitted to the system in real time through a smart gateway, and in this process, the representational state transfer (REST) API was used to enable communication with a real-time monitoring server. The server sends the processing results or the control and setting values of dehumidifying dryers to the smart gateway, and the gateway sends the values to the corresponding dehumidifying dryer. This study analyzed requirements for sharing data between dryers and server, and designed architecture to deliver a combination of synchronous/asynchronous processing models based on the plant operation environment and real-time data to the dashboard, where its operational feasibility was verified through experiment in an actual plastic manufacturing plant environment.

2. Smart Factory

In manufacturing plants in the past, humans managed the equipment, so equipment inspection occurred periodically or when problems with the equipment arose. This caused frequent suspensions of equipment use and led to a decline in the production rate or frequent replacement of equipment components. As a result, unnecessary costs occurred in manufacturing plants, and to solve this problem, technology to inspect equipment status in real time and predict problems with minimum labor was required. With the rise of IT convergence technology, smart factory technology that combines IT technology began to emerge in the manufacturing industry as well [8]. The smart factory is defined as a factory where physical production processes and operations are combined with digital technology, smart computing and big data to create a more opportunistic system for companies that focus on manufacturing and supply chain management. The smart factory is an aspect of Industry 4.0, a new phase in the Industrial Revolution that focuses heavily on real-time data, embedded sensors, connectivity, automation, and machine learning[11, 12]. As factories evolve in light of the data revolution, businesses need to rethink how they handle everything from automation strategies to workforce development tactics. Along the way, manufacturers will need modernized tools, including robust, flexible enterprise resource planning systems as a data and transactional backbone, that help them adapt quickly as they build toward a smart-factory future [13, 14]. Main focus areas in the smart factory are "PLAN, EXECUTE, MONITOR, ANALYZE". In this paper, we focus on monitoring area and propose the remote management system for smart factory.

3. Methodology

3.1 System Design

In traditional manufacturing plants, person manipulate equipment, check their status, and when an error occurs, they use a serial communication terminal to stop, check, and restart. It was also necessary to enter values and other preferred values depending on the raw material being processed through the connected terminals. Because all processes are managed by humans, it was difficult to handle errors immediately and detect and handle errors accurately. To overcome this, equipment needs technology that can grasp information in real time, detect and handle risks in advance, or respond immediately to errors.

Normal manufacturing plants operate multiple equipment, and when data from multiple equipment are shared with a real-time monitoring server, it is difficult to process synchronous mode requests and may cause a data bottleneck due to the nature of the factory operation environment. To solve this problem, data must be gathered and delivered through a gateway or data from the server should be delivered to the equipment. To handle such requests, a system architecture was designed, as shown in Figure 1. An IoT sensor was attached to each equipment to measure data from the equipment, and this data is transmitted to the connected smart gateway. The smart gateway collects measurement data from multiple equipment and delivers it to the real-time monitoring server. The monitoring server saves the measurement data from the gateway to the database in real time and displays a web-based dashboard based on accumulated measurement data. Also, it sends commands to control the equipment where errors are anticipated with the analysis of accumulated data.

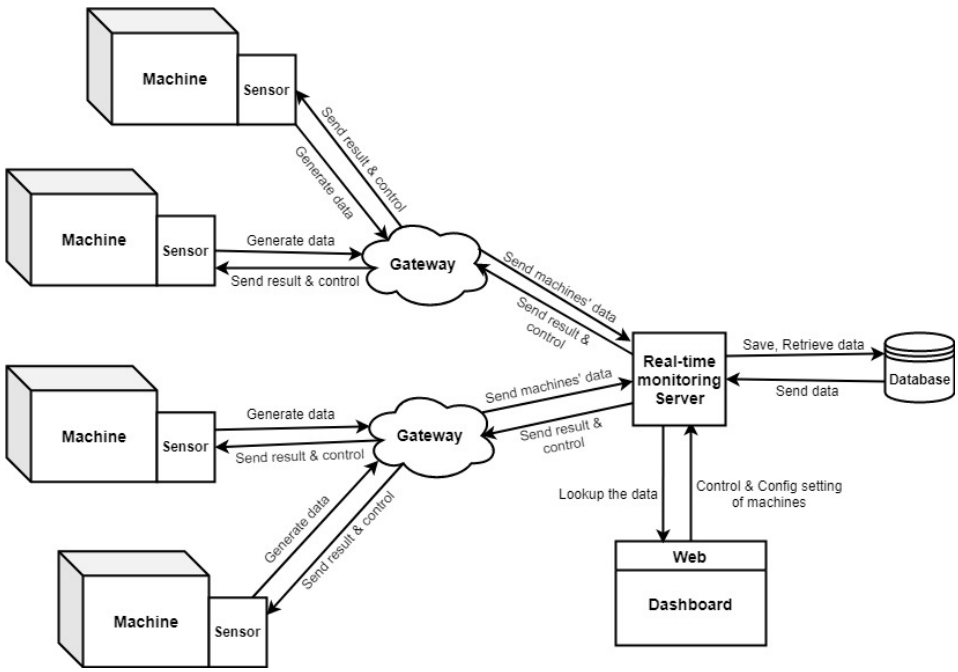

Figure 1. System architecture

Communication between equipment and the real-time monitoring server doesn't occur directly, but through a smart gateway using the REST API. The web-based dashboard in the real-time monitoring server is based on Socket.io to communicate with the server in real time, and the data that occur in the equipment can be checked through the web dashboard. We designed data store area to use two types of database one is for general management data and one for the cache. Figure 2 shows the detailed system design of the real-time monitoring server. Measurement data storage API occurs constantly as the equipment processes

products, and a caching technique was used with Redis to reduce database load, and all data were processed to be saved to the MariaDB.

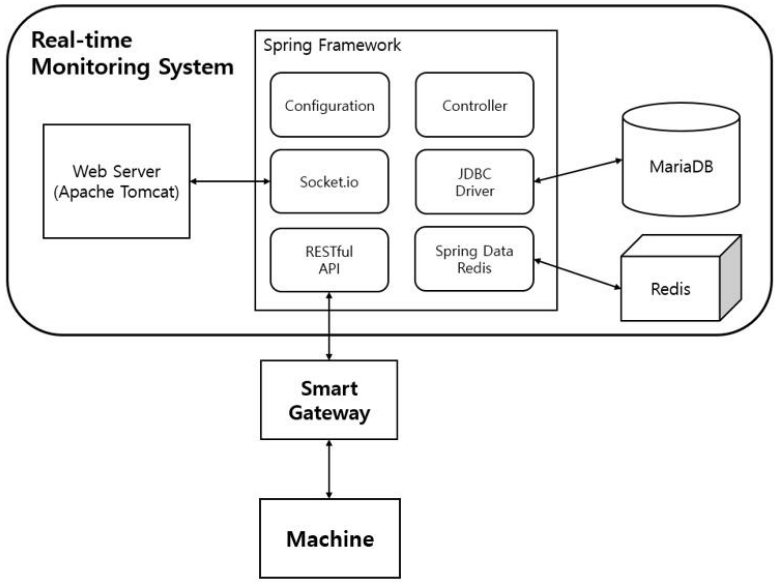

Figure 2. System design of real-time monitoring system

Table 1 shows a list of the REST API that is used for communication between the smart gateway and real-time monitoring server. The API's requests consist of equipment register, management, data, log and control, and a description of each request is as follows.

- Register: API used in the process of registering gateway and equipment to monitoring server
- Management: API that saves status and control information that occurs in the equipment operation process
- Data: API that saves the measurement information of products that are processed inside the equipment
- Log: API that saves event or error data during equipment operation
- Control: API used to control equipment remotely in a real-time monitoring server

Table 1. List of REST API

Request	URI	Description
Register (GET)	/data/reg/gw	Register gateway
	/data/reg/dry	Register machine
	/data/reg/confirm	Confirm connection between gateway and machine
	/data/drysvc/error	Send gateway's IP address if machine can't connect with gateway
Management (GET)	/manager/save/metaData	Save meta-data of machine
Data (GET)	/data/measureData	Save real-time measurement data
Log (GET)	/svc/save/eventLog	Save error or event log of machine or gateway
Control (POST)	/manager/control/setting	Send control order to machine

Figure 3 shows the database ERD schema of a real-time monitoring server. Descriptions of each table are as follows.

• dry_machine: A table that saves equipment status and operation information
• gateway: A table that saves smart gateway status or operation information
• measure_data: A table that saves data measured by an IoT sensor on the equipment, and holds information on products being processed. It delivers the equipment's ID value to the real-time monitoring server through the gateway to determine which equipment that data occurred from, so the ID value of the gateway that is connected to the relevant equipment is set as a foreign key.
• event_log: A table that contains various event logs from equipment or the gateway. It includes event logs from the factory's equipment or gateway operation, or log information on the error value of raw material that occurred in raw material processing.
• board: A table to display information from a real-time monitoring server's web dashboard to employees in the manufacturing plant
• user: A table to save user information in a real-time monitoring server's web dashboard

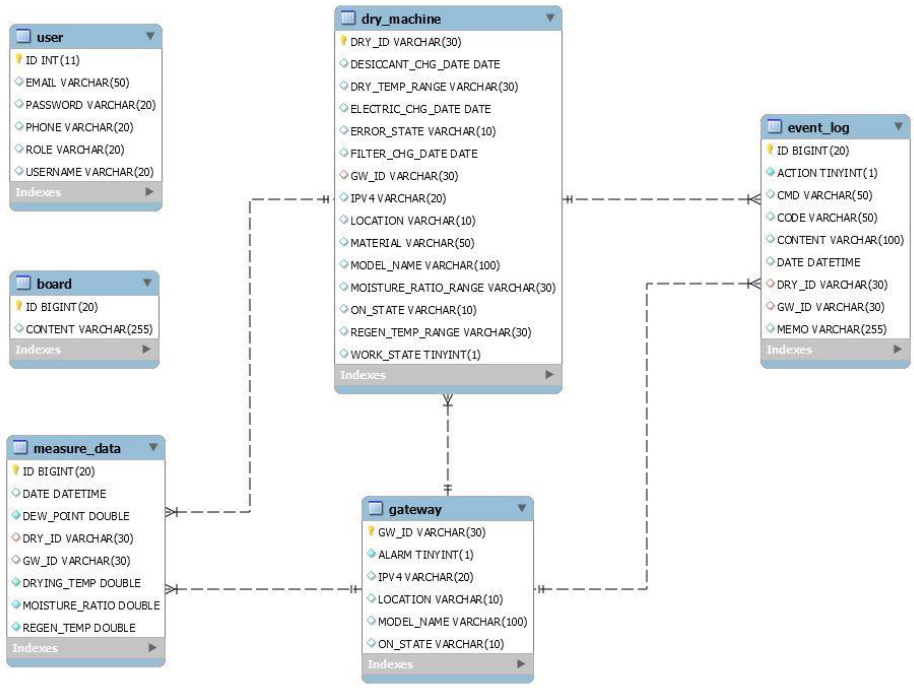

Figure 3. System design of real-time monitoring system

3.2 Software Design

3.2.1 Class Diagram

Figure 4 shows a class diagram of a web controller and REST API controller. The diagram is designed around a controller and entity, and the web controller is designed based on the MVC pattern and consists of classes for users' search efficiency in a web-based dashboard. The REST API is connected to objects that are used to communicate data with the gateway and is also connected to the "MeasureDataRedis" object for caching of measured data.

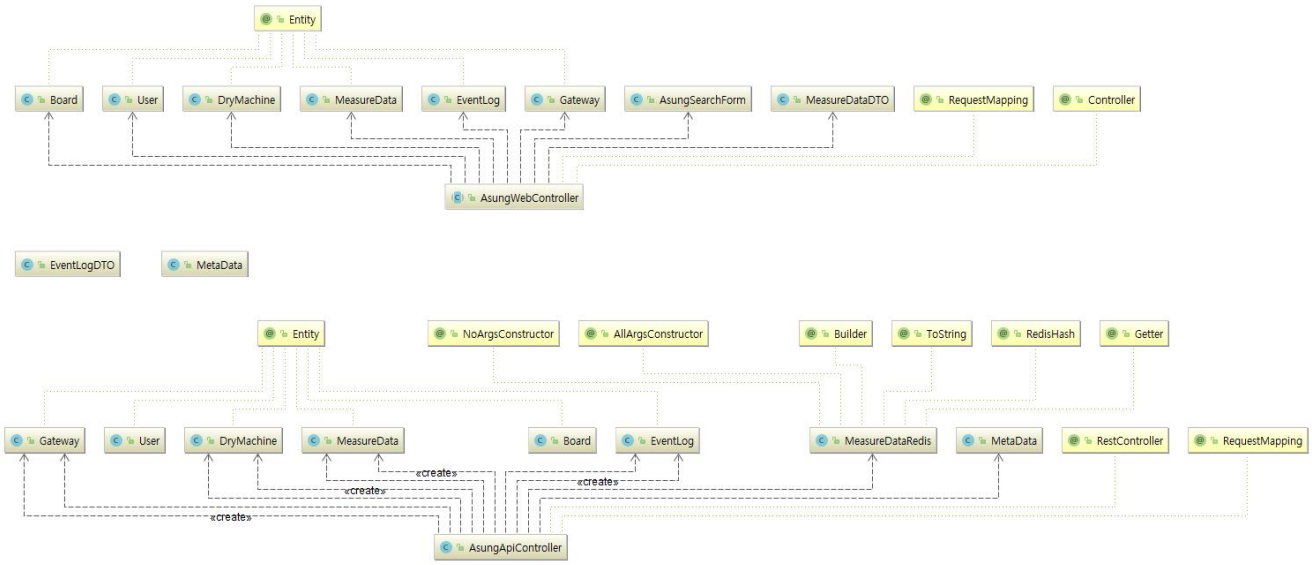

Figure 4. Class diagram of API and web controlles

Figure 5 shows a class diagram design regarding configuration. For API security, each registered smart gateway can block unauthorized API requests by using a registered JSON Web Token (JWT) as the key with OAuth2.0 authentication. Also, the basic configuration of Redis, which is used as cache storage for measured data in the server, is processed through “EmbeddedRedisConfig” class.

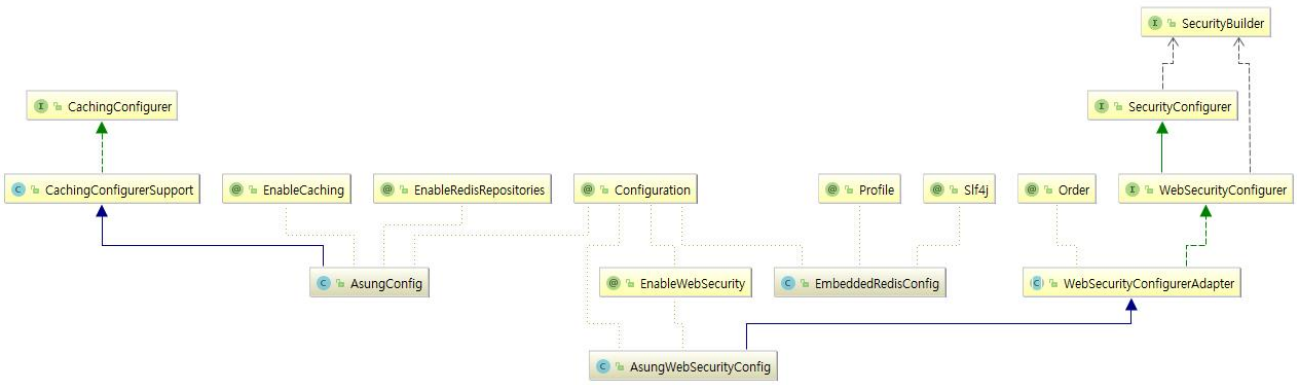

Figure 5. Class diagram of configuration

The database was connected using the Java Persistence API (JPA) and was implemented using Hibernate, which was basically used in the Spring Framework. Figure 6 shows a class diagram of the database connection . Some requests are not implemented through a standard JPA interface, where in this case, a separate Data Transfer Object (DTO) class was set apart to write JPQL and solve the problems.

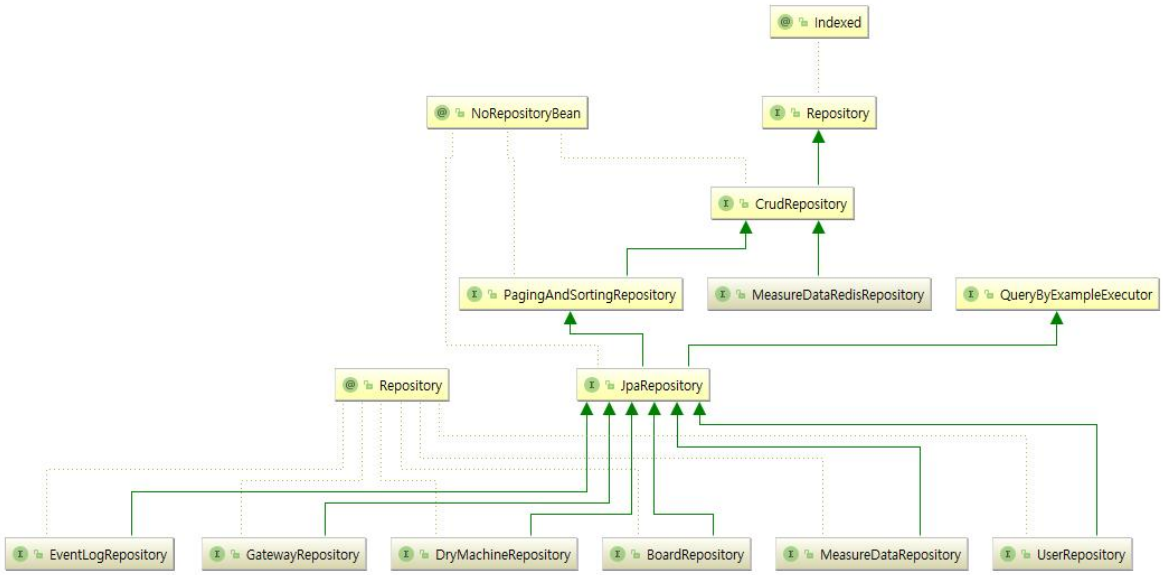

Figure 6. Class diagram of database connection

Figure 7 shows the entire execution process of a web-based dashboard. After log-in as administrator, the information saved in the database can be viewed as a chart, table or graph on each details screen, or the equipment configuration value can be changed remotely on the web through the machine management screen.

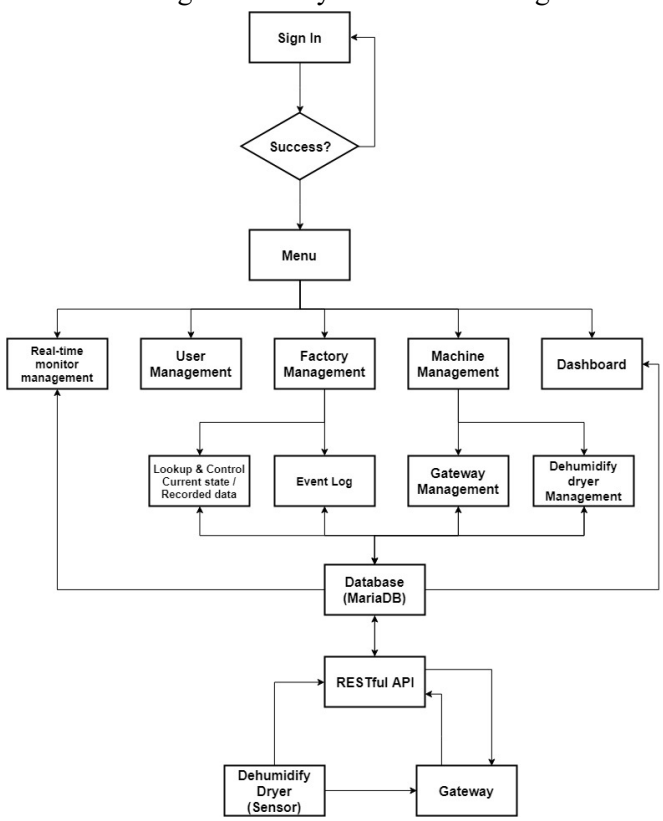

Figure 7. Process of administrator's web dashboard

4. Implementation and experiment

In this paper we designed and implemented a real-time monitoring system for smart factories, that is, for plastic manufacturing factories in the manufacturing industry. The experiment was conducted on a plastic manufacturing plant in the manufacturing industry. Figure 8 show the experiment environment. In plastic manufacturing plants, dehumidifying dryers are operated in the injection molding process, and the existing old dehumidifying dryers and an IoT sensor were used to design an environment to communicate with a server and gateway, and a Raspberry Pi-based smart gateway and real-time monitoring server were built.

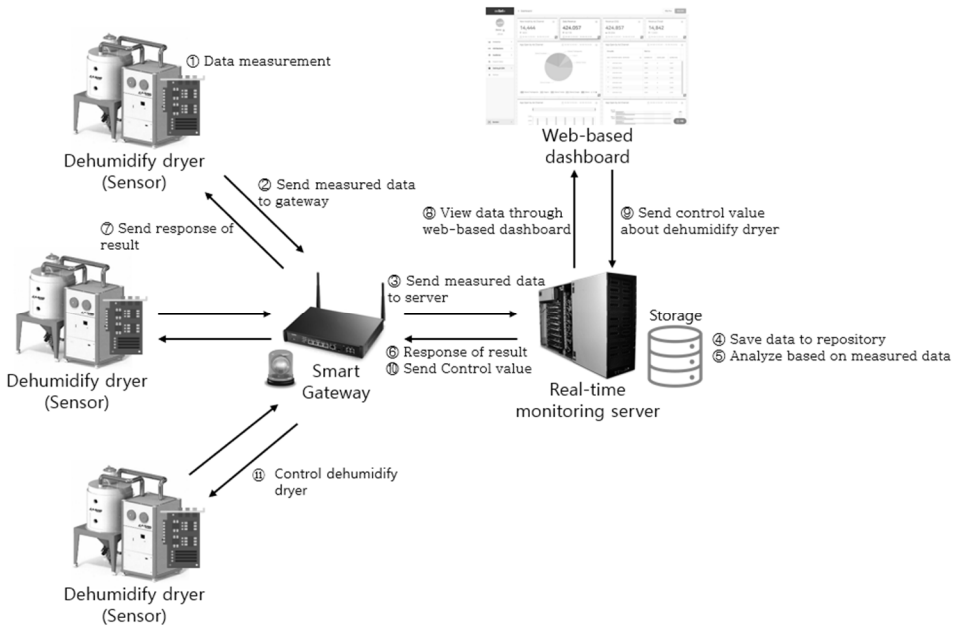

Figure 8. Experiment environment

The experiment environment is design for 3 dehumidify dryer and one gateway and the configuration which is consider for small plastic manufacture factory. Also, this configuration can be expanding to larger scale environment.

The dashboard was implemented with Spring boot, Thymeleaf, and Bootstrap. And socket.io was used to check the factory status in real time. In the real-time monitoring system, registration information and measurement data gathered from equipment sensors and the smart gateway are saved to the database through the REST API, and the saved data are displayed on a web-based dashboard screen. Figure 9 shows a web-based dashboard screen that was created according to the design.

(**a**) real-time monitoring screen (**b**) web dash board

Figure 9. Web dashboard screen

Figure 9a is the screen on the operation status and condition of all dehumidifying dryers seen on the large monitor in the factory. Since the factory is big and there are many machines, this screen can easily show equipment errors or operation status without checking the machines. Also, the real-time notice at the top of the screen displays announcements for the employees. Figure 9b is the first screen that appears after logging into the real-time monitoring server's web-based dashboard. It gathers event data and displays a donut chart on up to five events with the most frequent errors, so the administrator can focus on such event. Also, it shows the entire factory and equipment operation status, as well as recent measurement data, and provides monitoring of the expiration date of consumables used in dehumidifying dryers and events that occurred during the day.

Figure 10 shows the screen that provides the log of the API that is used in the operation of the real-time monitoring system. If communication with the gateway using the REST API is disconnected due to an error, the cause needs to be determined, so the API log is saved in the database to handle promptly in case of communication problems. Also, a search function based on error code, date, and HTTP method was added to find the API log of the desired type and period. Clicking the View button displays detailed information on the relevant APIs.

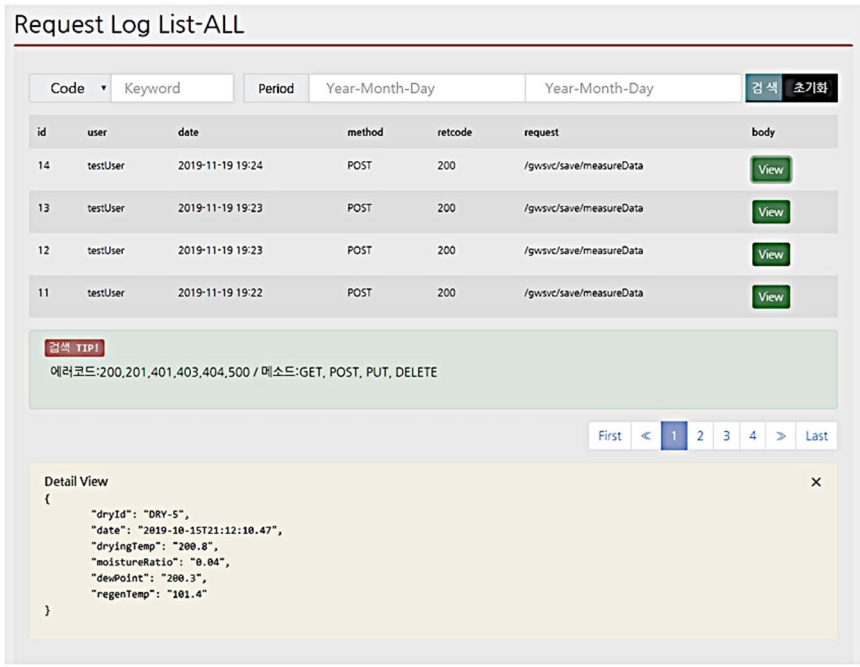

Figure 10. API request log

A real-time monitoring system that enables remote verification and control of product processing progress and equipment status was implemented in a manufacturing plant based on the Spring Framework. The real-time monitoring system and smart gateway communicate through the REST API, and the equipment data transmitted from the gateway can be saved to the real-time monitoring system's MariaDB. Also, equipment continuously measures data on the progress of product processing, and the factory operates multiple equipment for when database overload occurs in storing this data. For this, a Redis-based caching technique was used to save the equipment measurement data that occur frequently in the memory and save it to the database.

In a general manufacturing plant environment, the product processing status that is closely related to the product defect rate and factory's profit should be determined promptly and accurately. For this reason, product processing status is frequently measured, and this API occurs the most. In a plastic manufacturing plant, data measuring the raw material processing progress inside dehumidifying dryers occurs the most, and is an important element that determines defects in raw material, so a comparative analysis experiment was conducted on API processing speed when only the MariaDB was used, and when a Redis-based cache used on the MariaDB. Although the experiment was conducted in a plastic manufacturing plant, a simulation environment was created according to data and equipment number changes to verify that it can be operated in different manufacturing plant environments, and the experiment was conducted on API processing speed under the assumption that 30 measurement data storage API occurred per minute.

Table 2 shows the number of equipment, data, and equipment operation time by experiment. Experiment 1 compared the processing performance of real-time monitoring servers on measurement data storage API requests in consideration of changes in the amount of equipment. The experiment was conducted under the assumption that one gateway can be connected to 10 machines, and the actual number of machines and gateways was increased in the actual plastic manufacturing plant environment during the experiment. For a fair experiment, each detailed experiment was operated for 2 hours, and the average processing time for API requests was analyzed.

Table 2. Composition of experiments

	Number of dehumidifying dryers	Number of gateways	Running time (hour)	Total requests
Experiment 1	1~100	1~10	Two for each condition	36,000~360,000
Experiment 2	20	2	5	180,000

Figure 11 shows a graph of average processing speed in Experiment 1. When the number of machines was small, the average processing speed was high, but as the number of machines increased, processing speed maintained a certain level. Also, combining the caching technique using Redis showed a faster average processing speed than using the MariaDB alone, which shows that applying a Redis-based caching technique to frequently used API requests can process it more efficiently regardless of the increase in the number of equipment.

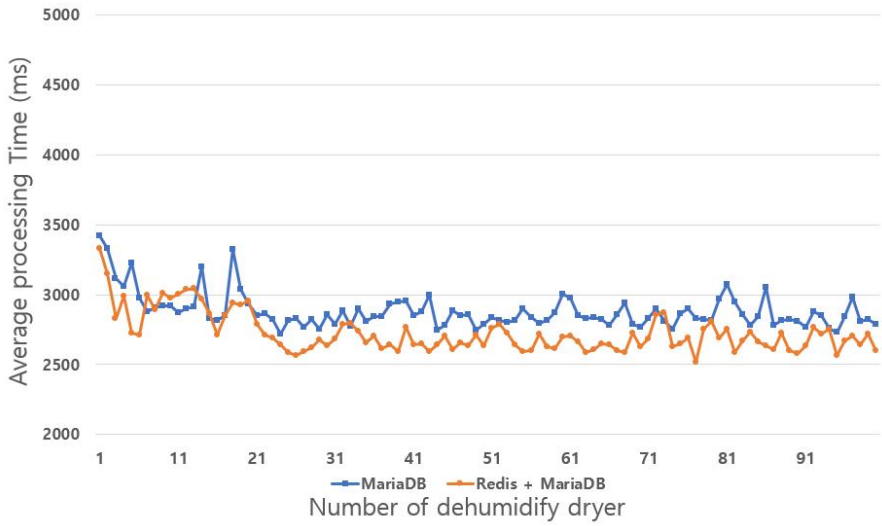

Figure 11. Average procession speed as gateways and dehumidifying dryers increase

Experiment 2 compared the processing performance of measurement data API requests when the equipment was operated for 5 hours, considering that manufacturing plants are operated for long hours. The number of equipment was fixed at 20 and gateways at two, and they were operated for 5 hours simultaneously while the average processing speed of API requests from each equipment by the real-time monitoring server was analyzed. Figure 12 is a graph that shows the average processing speed in Experiment 2. In the early stage of equipment operation, using the MariaDB alone and applying the Redis-based caching technique didn't show a big difference. However, as time passed and similar types of API requests were delivered to the real-time monitoring server, the Redis-based caching technique had better average processing speed. This shows that when equipment is operated for many hours in a manufacturing plant, using Redis with a cache to process API requests is more efficient.

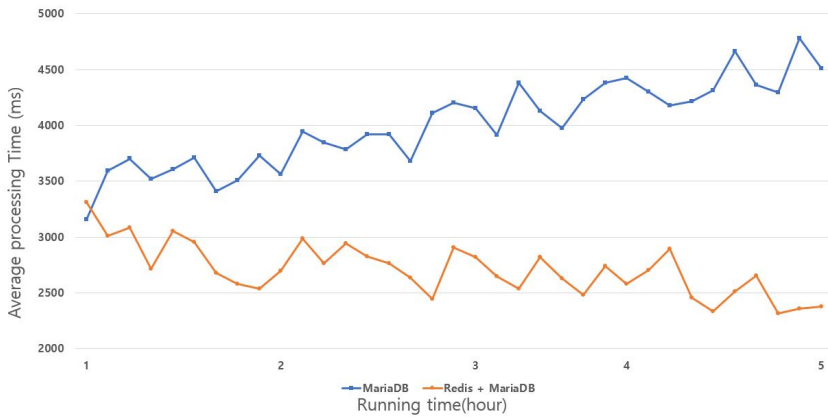

Figure 12. Processing speed as running time increase

4. Conclusion

In this study we designed and implemented a real-time management system, which is essential in using

existing factory equipment and realizing a smart factory in manufacturing plants that have difficulties to equipped advanced equipment because of cost limitations. Among various fields of the manufacturing industry, an experiment was conducted with a plastic manufacturing plant, where an IoT sensor was attached to dehumidifying dryers, which play the most important role in plastic manufacturing plants, and a real-time monitoring system that communicates to such a sensor was implemented. Also, to solve database overload that can occur due to massive data in the factory operation process, a Redis-based cache was used, and the experiment in an actual factory environment confirmed that it offered better performance than using the database alone. Through the experiment we could show efficient way to implement remote management system for smart factory operation with real-time monitoring and remote control in an existing system. Also, our system can be operated in various environments in addition to plastic manufacturing plants.

Acknowledgement

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2020-2017-0-01630) supervised by the IITP(Institute for Information & communications Technology Promotion).

References

[1] B.M. Mun, M. Lim, S.J. Kim, S.J. Bae, Fault Detection and Diagnosis of Smart Factory Equipment Using Wavelet Spectrum, Korean Reliability Soc., Vol. 19, pp 22-30, Mar. 2019.
DOI: https://doi.org/10.33162/JAR.2019.03.19.1.22

[2] S.H, Moon, Analysis of ICT Converged Smart Factory and its Driving Strategy, The Journal of the Convergence on Culture Technology, Vol. 4, pp. 235-140, Aug. 2018.
DOI: http://dx.doi.org/10.17703/JCCT.2018.4.3.235

[3] F.M. Tigao, F. Paula, Review on the Application of Blockchain to the Next Generation of Cybersecure Industry 4.0 Smart Factories, IEEE Access, Vol. 7, pp 45201-45218, April 2019.
DOI: https://doi.org/10.1109/ACCESS.2019.2908780

[4] Y. Lee, W. Lee, S. Lee, The Development of Protocol for Construction of Smart Factory, Inst. Korean Electr. Electron. Eng. Vol. 23, pp 1096-1099, Sept. 2019.
DOI: https://doi.org/10.7471/ikeee.2019.23.3.1096

[5] M. Ahmed, Q. Zhu, S. Afsar, IoT based real time agriculture farming, International Journal of Advanced Smart Convergence, Vol. 8, pp. 16-25, 2019.
DOI: http://dx.doi.org/10.7236/IJASC.2019.8.4.16

[6] B, Chen, J. Wan, Lei. Shu, P. Li, M. Mukherjee, B. Yin, Smart Factory of Industry 4.0: Key Technologies, Application Case, and Challenges. IEEE Access, Vol. 6, pp 6505-6519, Dec. 2017.
DOI: https://doi.org/10.1109/ACCESS.2017.2783682

[7] M.H. Song, A Study on IoT based Real-Time Plants Growth Monitoring for Smart Garden, International Journal of Internet, Broadcasting and Communication, Vol. 12, pp. 130-136, 2020.
DOI: http://dx.doi.org/10.7236/IJIBC.2020.12.1.130

[8] H. Lee, Embedded System Framework and Its Implementation for Device-to-Device Intelligent Communication of Manufacturing IoT Device considering Smart Factory, Journal of Korean Institute of Intelligent System, Vol. 27, pp 459-465, October 2017.
DOI: https://doi.org/10.5391/JKIIS.2017.27.5.459

[9] Y. Choi, I. Park, D. Yang, B. Ha, M. Heo, J. Lee, A Study on the Characteristics of Plastic Injection Molding Using Core in Core Cooling Technology. J. Korean Soc. Manuf. Proc. Eng., Vol 18, pp 82-87, Mar. 2019.
DOI: https://doi.org/10.14775/ksmpe.2019.18.3.082

[10] J. Shim, J. Lee, Convergence Security Technology of OPC-UA Protocol Gateway based on DPI & Self-Similarity for Smart Factory Network, Korea Inst. Inform. Security and Cryptology. Vol. 26, pp 1305-1311, 2016.
DOI: https://doi.org/10.13089/JKIISC.2016.26.5.1305

[11] D. Santos, J. C. Ferreira, IoT Power Monitoring System for Smart Environments. Journal of Sustainability. Vol. 11, pp 5355, 2019.
DOI: https://doi.org/10.3390/su11195355

[12] Y. Lee, W. Lee, S. Lee, Development of Embedded Board for Construction of Smart Factory. Inst. Korean Electronic Engineering. Vol. 23, pp 1092-1095, 2019.
DOI: https://doi.org/10.7471/ikeee.2019.23.3.1092

[13] Abas ERP Group, Smart Factory Manufacturing, https://abas-erp.com/en/news/smart-factory-manufacturing

[14] S.Y. Hwang, D.J. Shin, K.J. Kwak, J.J. Kim, J.M. Park, Real-time Processing of Manufacturing Facility Data based on Big Data for Smart-Factory, The journal of the institute of Internet, Broadcasting and Communication, Vol 19. pp. 219-227, 2019.
DOI: https://doi.org/10.7236/JIIBC.2019.19.5.219