

Container-Based Multi-purpose IoT Architecture for User-Friendly Applications with Cloud Chatbot Agent

Jaehyeong Lee, Changyong Um, Soonsung Hwang, and Jongpil Jeong(✉)

Department of Smart Factory Convergence,
Sungkyunkwan University, Suwon, Gyeonggi-do 16419, Republic of Korea
{objective,e7217,suns303,jpjeong}@skku.edu

Abstract. IoT is a new era technology that can be used by end users because of its functionality and necessity. Most of IoT devices are aimed at versatile smart devices, and these features need to be conveniently provided to users, such as for use in mobile. In addition, these devices must be able to tailor various functions and purposes to the needs of the user. Currently, open source platform provides flexibility in perspective of the developers in order to enable multi-purpose as a single device in terms of requirements for end users. However, this flexibility should be reconsidered from the end users, too. In this paper, we propose the cloud based multi-purpose IoT and provide convenience to the end user through the chatbot agent for end user friendliness. In addition, container technology is applied for versatility and flexibility of devices. These researches can be useful in multi-purpose robot pendants, protocol conversion genders in factory area, and motor boxes that can convert gears in smart toys situations where change flexibility and affinity are required according to the request of users.

Keywords: Internet of Things · User-Friendly Applications · Chatbot agent · Docker

1 Introduction

IoT is creating a new digital age as a mega-trend. IoT, which is different from existing web technology, has new value by connecting objects and interacting and interacting with the Internet [1,2]. In the IoT environment, the intelligent 'Things' have the purpose of being able to clearly recognize the state of the surroundings by continuously providing their own state, making it easy to make the decision making [2]. IoT is not just a field, it is changing the whole daily life and it is being applied and developed all over the industry. According to the Gartner report and Cisco's announcement, more than 2 billion connected products and over 50 billion smart objects are expected to emerge by 2020, and this is realizing [3].

© Springer Nature Switzerland AG 2019
S. Misra et al. (Eds.): ICCSA 2019, LNCS 11624, pp. 121–134, 2019.
https://doi.org/10.1007/978-3-030-24311-1_9

For using IoT more clearly, it is necessary to study about user-friendly human to machine interface (HMI) [4]. Historically, the most effective HMI is smart phone. Most modern people have a smart phone and this personalized end device, is being used to connect people and people. For human-to-human communication, voice transmission, e-mail, and instance message service applications are commonly used, which can be a good model for IoT. According to this necessity, in the IoT environment, it has been developed to use chatbot of instance message application [5]. The chatbots in the IoT environment are similar to the agent concept. Agent generally means "self-contained, interactive, and simultaneously running object" that has internal state and communication functions [6].

Docker, a Linux-based container technology, has a platform for both distributors and developers, making it easy to deploy and develop. In addition, it is possible to execute an application in an environment independent manner, and it is possible that these applications do not influence each other [7].

As mentioned above, chatbot can perform enough role as an agent for end users in IoT environment, and it is possible to customize IoT affinity through them. Also, using Docker as a the container technology, one device can be used for multi purpose according to the needs of the end user. Therefore, the use category of chatbot for multi-purpose device in IoT environment can be broadened when using Docker, and the affinity of platform can be improved. For example, robot control pendants commonly used in smart factories are all provided as dedicated controllers for the robot itself. However, the IoT environment can present a model through a multi-purpose IoT, such as a single stand-alone model and connecting through an application in it.

User-friendliness in software and architecture is very important. User friendliness is a broad representation of a user's experience and can be measured in terms of usability in terms of software and architecture. This is a key indicator of how easy it is for users to use the software and adapt to the user's various requirements. The existing models for usability measurements are represented by the Nelson model and the ISO 9261 model, which is sufficient to evaluate the architecture and software quality. Therefore, we can discuss the user-friendly application of Cloud based IoT.

In this paper, we propose a new method to improve the end user affinity of cloud based multi-purpose IoT by combining chatbot and container technology. This paper has contributed by convergence of parts and presenting a new type of architecture and device. Also, by evaluating each model as user friendliness, we express the advantage of this system.

This paper is configured as follows: In Sect. 2, we discuss the multi-purpose IoT, chatbot, Docker, and model for evaluating user-friendliness. In Sect. 3, we combine the Docker and chatbot agent for cloud based multi-purpose IoT and describe the overall architecture. In Sect. 4, we evaluate this architecture through a model for implementing the entire system and evaluating user-friendliness. In Sect. 5, we discuss the contribution of this paper and future research.

2 Related Work

2.1 Cloud Based Multi-purpose IoT

IoT is a global infrastructure for the information society, providing improved services by interconnecting things based on existing evolving, interoperable information and communication technologies (ICT) [8]. The environment with numerous devices and information objects is connected through the network. The vision of loT is a smart environment that is all connected by smart devices. Using IoT, you can seamlessly integrate physical objects with Internet-like systems, allowing you to interact with physical objects and connect to cyber agents to achieve mission-critical goals [9,10].

Cloud computing is a computing paradigm that provides users with an on demand ICT infrastructure. This includes servers and networks where users can access information, which can be connected anytime and anywhere at the request of the user. This model eliminates the need to physically store and process data in the same space by providing the virtual space in which data of user is stored and processed [11,12].

Cloud-based IoT is a model that improves the efficiency of work by solving tasks that require complex computation or high computing power in the cloud in order to solve resource constraints of devices that can generally perform limited computing. Cloud-based multi-purpose IoT performs data storage and processing in the virtual space of the cloud. In addition, it can sensitively respond to user requests and actively change its functionality within limited device resources. Therefore, user friendliness and agility to changes become essential requirements.

2.2 Chatbot Agent

Chatbot is also called messenger bot because it is based on text messages that user and bot exchange in the chat space. The user enters a text message in the virtual space, and the software analyzes the message, creates an appropriate response, and sends the message back to the user. Therefore, the user can communicate with the soft-agent-bot through the interaction with the soft-agent-bot and confirm that the soft-agent-bot has performed the correct operation. In general, chatbot is a system that send corresponding messages according to the rules that are already defined as needed. In the early developed version, a method of confirming simple matching of keywords and returning a value has been performed. However, due to recent developments of various technologies, a natural language entered by a user is grasped and analyzed to define requirements. Chatbot can perform the requirements and respond to the results [13].

The function of such a chatbot means that the chatbot can operate as an agent as mentioned above. Agent is an abstract object in terms of software engineering, and has the same domain as methods, functions, and modules, but differs in that it performs autonomous operations instead of simple functions [6,14]. The chatbot agent will have the key technologies listed in Table 1.

Table 1. Chatbot agent technology.

Function	Description
Natural Language Processing	Ability to recognize and process human language Features such as search, query, and translation
Semantic Response	Responsive technique that can perform logical estimation and action using recognized meanings
Text Mining	Ability to acquire new data from unstructured text data
Context-aware Computing	Technology that provides user-centered intelligent service that informs current situation

2.3 Docker: Linux Light-Weight Container

Light-weight virtualization technology can reduce the amount of computing resources required to run guest OS compared to virtual machine (VM). During the past hour, server virtualization and isolation have been implemented using hypervisor-based virtualization technology in general [15]. Customize virtual hardware and virtual applications to create a virtual machine that is independent of the underlying host. The operating system is installed and works on each VM instance. Container virtualization is different from other virtualization technologies in terms of virtualization and quarantine, but it can be implemented functionally close. This can be considered a viable alternative to hypervisor-based virtualization [16].

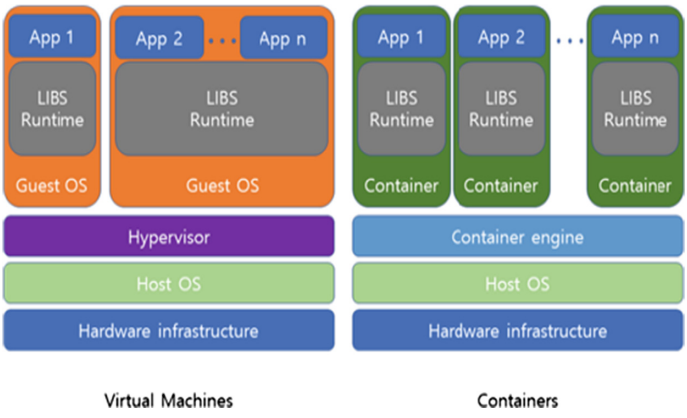


Fig. 1. Virtual machine and container.

Figure 1 shows the types of virtual machines and container-based structures. Unlike hypervisor-based virtual machines, container-based virtualization implements processes through container engines. OS-level process isolation does not

require hardware and driver virtualization [16]. In particular, containers share the same OS kernel as the underlying host system and provide independent characteristics of the virtual network interface, process space, and file system. These shared kernel features enable containers to reduce image volume and enable more efficient virtualization systems.

It is the Docker platform that makes container concepts more widely used. Docker leverages the underlying container for engines and provides the ability to efficiently create, run, manage and remove containerized applications. A Docker container is created based on a base image stored in a particular private or public registry. The Docker can add a read/write layer on top of the image and store the image in a hierarchical form. Also, Docker has a fast implementation process. Lightweight properties of containers can be utilized in various networking areas. For example, an under-processing device, such as a single-board computer, can efficiently run container-based applications based on container images associated with IoT edge computing. Virtualization technologies such as VMs, containers, and Unicorns have been actively studied as an important element of the architecture for implementing IoT services [17,18].

2.4 User-Friendliness: Usability Model

User friendliness is a factor that can express the user's experience and can be measured by usability. Usability is a conceptual indicator to confirm convenience, ease in using software products. Since software has begun to be used, this factor has been valued as a very important value. In this area, Nelson model and ISO model are mainly used [19]. Software in the Nielsen model has been studied as an attribute that influences software acceptability as a single product. He has divided this acceptability into practical or social acceptance and has come up with an assessment of usability through the provision of five sub-attributes: learnability, efficiency, memorability, errors, and satisfaction. Another model that is often used is the ISO 9216 model(2001). The International Standardization

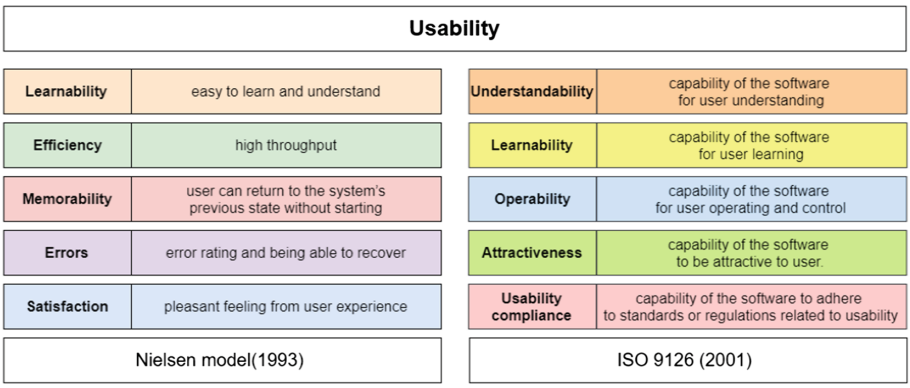


Fig. 2. Usability models.

Organization defined sub-attributes of usability as understandability, learnability, operability, attractiveness, usability compliance. The definitions of these two models are shown in the following Fig. 2.

3 Container-Based Multi-purpose IoT Architecture for User-Friendly Applications

We propose a user-friendly application for multi-purpose IoT, which is the main aim of this paper. To this section, we propose a new system architecture, demonstrate the user-friendliness, agility, and flexibility that this model can have by experimenting with connecting Docker and Telegram chatbot to actual IoT.

3.1 System Architecture

Figure 3 shows the architecture for rapidly changing the multi-purpose IoT according to the user's request while using the user-friendly chatbot. Multi-purpose IoT has a lot of applications that are being held for versatility. Also, it has contained client to connect with cloud and to operate Docker agent. The client will connect to the chatbot server by connecting to the Internet after confirming the network at the first operation. You can also launch and delete the contained application in the future and download the Docker image from the cloud. In the cloud area, a chatbot server exists to provide virtual space to connect with IoT and Telegram users. Through this, an authentication function for mutual authentication and a database function for storing data generated

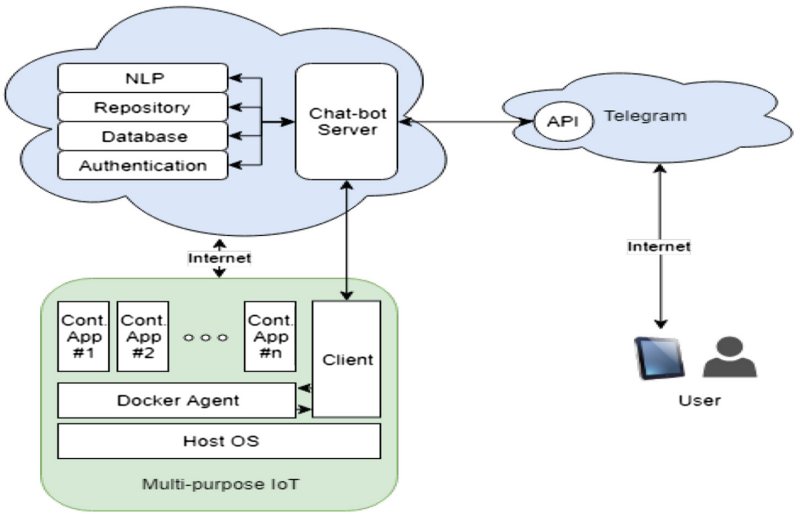


Fig. 3. System architecture.

in IoT and telegram should be provided. In addition, you should have a repository for linking Docker application images and a Natural Language Processing (NLP) processor that can process your language from Telegram messenger. The user can instruct the work through the process of transmitting text data to the chatbot through the telegram app. The job can be provided with a report on the current status of the device, the current status of the current Docker images, and the number of current Docker images. You can also interrupt the execution of the image of the locker to change the state of the device or command the download of the Docker image to run a new image of the Docker.

3.2 Application of Chatbot Agent

The chatbot agent uses the data received from the Telegram API and the data received from the IoT device to determine the state of the device. Assuming all available apps are Docker images, you can run the device by running this Docker image at your request. Under this assumption, the user can request the following requirements from the device.

- User requests authentication to use IoT device
  The user enters the serial number of device into the chatbot to register IoT device. If the device has been previously registered via the Internet, the user is granted the privilege through the serial number inquiry. In the telegram, since the user is identified with the telephone number, compliance status of the user is already secured. Therefore, there is no need for a different security for using IoT, and if possible, the IoT device and the user smart phone can be mutually authenticated through a device serial number and phone number exchange process at a short distance
- User inquires current device status and application behavior and status
  The user sends a status inquiry message to the chatbot through the Telegram. The chatbot agent processes the message in a natural language through the NLP and recognizes the request of user based on the message. The chatbot agent transmits the current device status inquiry to the client that has finally transmitted the data according to the request of the proper user registered, and the client transmits the status data to the client. The status data should include not only the status of the device but also whether or not the application is operating, whether it is present or not, and whether it operates normally. The chatbot agent transmits the inquired data through the Telegram API so that the user can check the status of the device and the operation of the application.
- The user can inquire the application which can be installed on the device or already installed, and download new one
  User can use the Telegram, server, NLP as mentioned above to query the currently possible installing and already installed applications. The chatbot agent introduces the application that exists in the repository (in Docker registry) based on the message, and inquires and delivers the installed app based on the data received from the client. If user want to install a new Docker

image according to the request, the chatbot server will send the address of the repository to the client and IoT will download the Docker image from the repository via the connected Internet.

- User want to delete, disable, and launch other apps
  If you want to delete the already installed Docker image according to the user's request, the chatbot server requests the client to stop and delete the image. This makes it easy to delete apps that users have already installed. Also, since it is already contained image, it has no effect on other operations of IoT. If the user is asked to stop the running app and run another app, the chatbot agent will also send the request to the client and run it.

4 Experimental Results

4.1 Experimental Environment and Method

We can see this by connecting the client applied to IoT and the chatbot agent in the public cloud area. In this experiment, NLP was excluded for the simplicity of the experiment. Raspberry pi 3B+ was used as an end device. Chatbot server was implemented in Ubuntu 18.04 LTS version, which is a laptop VM instance. In this experiment, we proposed an IoT device as a temperature and humidity gauging unit and used AM2315 sensor. The entire process of using the Docker and chatbot was very flexible and efficient because it used the existing open-source platform. In order to perform the operation of chatbot agent mentioned above, operation options of chatbot are specified as shown in the following Table 2.

Table 2. Chatbot command.

Command	Description
/status	Report current operation status of Docker image, existence of image and status of device
/run d-img 'name'	Docker image with name will be executed
/stop d-img 'name'	Docker image with name will be stopped
/delete d-img 'name'	Docker image with name will be deleted

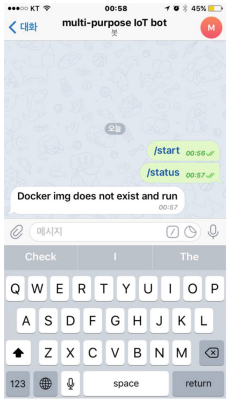
Raspberry Pi, an end device for operation, has also prepared the following Docker image to assume that various modules can be combined. In order for the Docker image to be executed, the Docker image is configured so that a run Docker agent is executed through the client and at the same time a specific file of the image is started. The configured Docker image is shown in Table 3.

The experimental procedure is as follows. Step 1. Docker image does not exist and current status inquiry, Step 2. After Img I download, executing Img I automatically, Step 3. Current status inquiry again, Step 4. Delete downloaded Img I, Step 5. After downloading Img II, executing Img II, and step 6. Stop Img II

Table 3. Docker images.

Image name	Description
Img I	Temperature measurement
Img II	Humidity measurement

When Step 1 is executed, we can see through the Telegram API that the server receives the message sent by the user and reports the current application status of IoT. The IoT client inquires its status according to the request of the server and transmits its status to the server again. Figure 4a and b show IoT status and return data in Telegram.



(a) Step 1 in Telegram



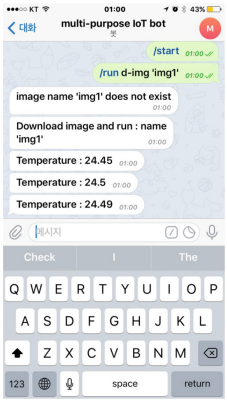
(b) Step 1 in IoT

Fig. 4. Experimental Step 1.

When Step 2 is executed, the IoT client was asked to run a specific image but did not have the image. Therefore, the IoT client automatically downloads and runs an image that it holds in the cloud repository. This automatic action is one of the main functions of the proposed system and the chatbot agent and shows that the end user can conveniently use it without any new connection or control of the developer. Figure 5a and b show the results.

Due to the result of Step 2 of the experiment, the 'Img1' Docker image was present in the IoT, and it was verified through the Telegram with Step 3. Figure 6a show these results. Also, by allowing the image to be deleted through the Telegram, the management of the IoT application with Step 4 was able to confirm the user friendliness through the Figs. 6b and 7a.

By performing the tests in Step 5 and Step 6, we can confirm that the process can be held in a halt state and that the application can be run again. Everything was done through Telegram and cloud-based IoT systems without downloading

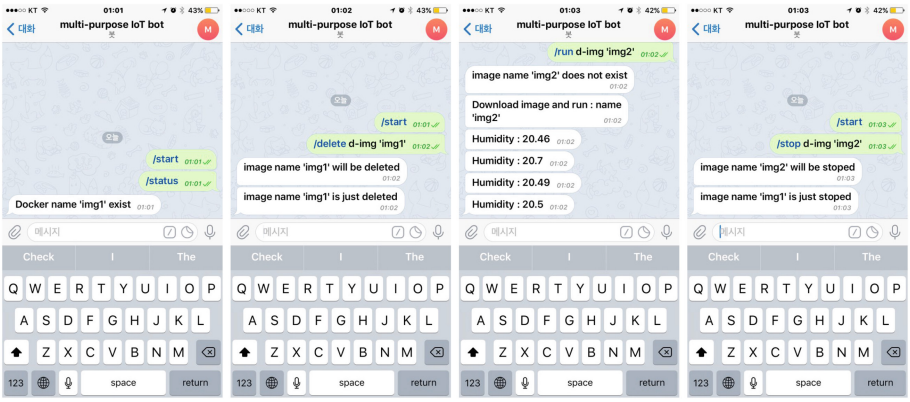


(a) Step 2 in Telegram



(b) Step 2 in IoT

Fig. 5. Experimental Step 2.



(a) Step 3 (b) Step 4 (c) Step 5 (d) Step 6

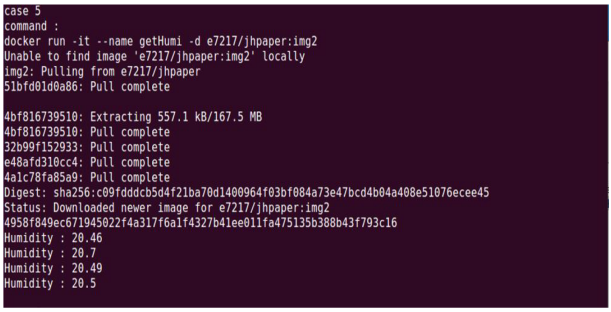
Fig. 6. Experimental Step 3, 4, 5 and 6 in Telegram.

new mobile applications. Figures 6c and 7b show the results of Step 5, while Figs. 6d and 7c show the results of Step 6.

Through these experimental results, we were able to guarantee a user friendly connection between the server and the IoT client that handles the message through the Telegram API. In addition, this method is possible to secure the multi-purpose IoT by running and downloading the image of the Docker and controlling the installed application.



(a) Step 4



(b) Step 5



(c) Step 6

Fig. 7. Experimental Step 4, 5 and 6 in IoT.

4.2 Evaluation of User-Friendliness

As already mentioned in Sect. 2.4, the model that evaluates the usability of the software generally use the Nelson model and the ISO model. We can evaluate the quality of the architecture by comparing the cases using the proposed architecture, using only the chatbot, and using only the Docker.

Evaluating the case that only the chatbot is used by the Nelson model and ISO increases the ease of handling by user and efficiency of the application of the IoT device through the chatbot and IoT server. However, if the application is downloaded through the server, It will be downloaded directly to the main OS system. There is a disadvantage in that it is fundamentally overlapping with other applications and can lead to errors. This exists as a factor that decreases the error in the Nelson model and the usability compliance in the ISO model. In addition, it is difficult to retrieve the state of the library already installed, and it is difficult to delete the libraries.

When evaluating the case where only the Docker is used, the user must meet the unfamiliar application user interface (UI) or without the UI like developers. In addition, there is a disadvantage in understanding or using it because it is inconvenient for recognizing the user's natural language.

When evaluating the use of a chatbot with Docker, ordinary users can control their own devices through familiar and familiar UI screens that they use every day, have. Moreover, the method of recognizing natural language through chatbot as well as the way through UI is progressing continuously. Also, even if you delete the program you are using, it does not affect the main OS because it is already containerized per application. In case of an error, it is easy to modify the Docker image, so there is ease of use and error management.

The overall evaluation of these assessments can be shown in the following three graphs through the Fig. 8.

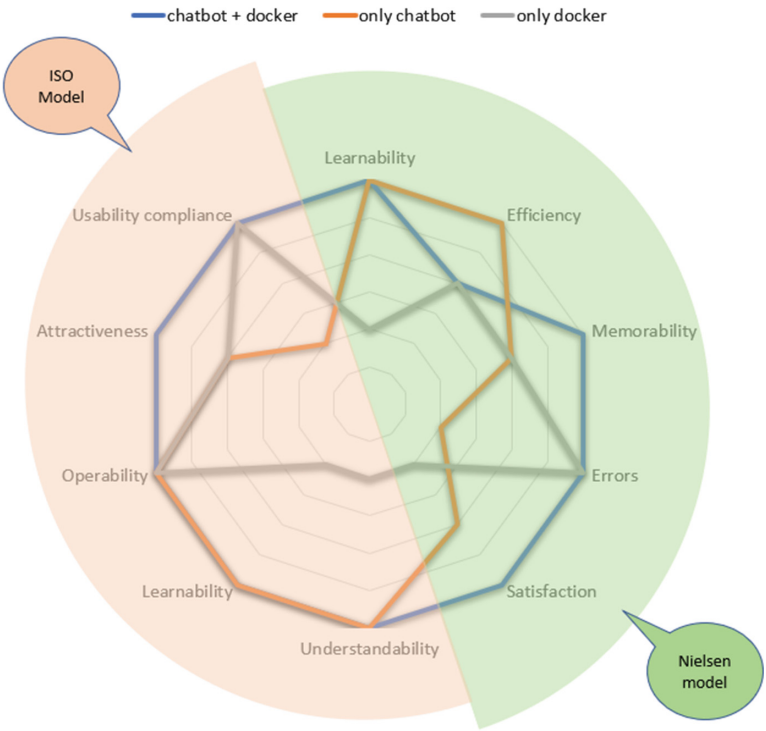


Fig. 8. Evaluation of User-friendliness.

5 Conclusion

In summary, this research is shown that it is possible to use a Docker image to control a multi-purpose device from a conventional research that controls a cloud based IoT through a message using a chatbot, and it is possible to change the device purpose according to a user request. We also confirmed that this whole process has a user friendness that developers do not need. Using these solutions, universal pendants for in-plant industrial robots can be produced independently

for multiple vendors and can be used for protocol conversion genders that need to be changed according to the situation. Future research will focus on user-friendly industrial platforms or user-friendly robot pendant using large-capacity text natural language processing using 5G.

Acknowledgment. This work has supported by the Gyeonggi Techno Park grant funded by the Gyeonggi-Do government (No. Y181802). This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2019-2018-0-01417) supervised by the IITP (Institute for Information & communications Technology Promotion).

References

1. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of Things (IoT): a vision, architectural elements, and future directions. Future Gen. Comput. Syst. **29**(7), 1645–1660 (2013)
2. Evans, D.: The Internet of Things How the Next Evolution of the Internet is Changing Everything. Cisco Internet Business Solutions Group (IBSG), pp. 346–360 (2011)
3. van der Meulen, R.: Gartner Says 6.4 Billion Connected 'Things' Will Be in Use in 2016, Up 30 Percent From 2015. Stamford, Connecticut (2016)
4. Ungurean, I., Gaitan, N.C., Gaitan, V.G.: An IoT architecture for things from industrial environment. In: 2014 10th International Conference on Communications (COMM), pp. 1–4 (2014)
5. Kar, R., Haldar, R.: Applying chatbots to the internet of things: opportunities and architectural elements. Int. J. Adv. Comput. Sci. Appl. **7**(11), 147–154 (2016)
6. Nwana, H.S.: Software agents: an overview. Knowl. Eng. Rev. **11**(3), 205–244 (1996)
7. Merkel, D.: Docker: lightweight Linux containers for consistent development and deployment. Linux J. **239**, 2 (2014)
8. Bandyopadhyay, D., Sen, J.: Internet of things: applications and challenges in technology and standardization. Wirel. Pers. Commun. **58**(1), 49–69 (2011)
9. Stankovic, J.A.: Research directions for the internet of things. IEEE Internet Things J. **1**(1), 3–7 (2014)
10. Santucci, G.: Internet of the future and internet of things: what is at stake and how are we getting prepared for them. In: eMatch Conference, Oslo (2009)
11. Rimal, B.P., Choi, E., Lumb, I.: A taxonomy and survey of cloud computing systems. In: 5th International Joint Conference on INC, IMS, and IDC, NCM 2009, pp. 44–51 (2009)
12. Duan, Y., Fu, G., Zhou, N., Sun, X., Narendra, N.C., Hu, B.: Everything as a service (XaaS) on the cloud: origins, current and future trends. In: 2015 IEEE 8th International Conference on Cloud Computing (CLOUD), pp. 621–628 (2015)
13. Noguera-Arnaldos, J.Á., Paredes-Valverde, M.A., Salas-Zárate, M.P., Rodríguez-García, M.Á., Valencia-García, R., Ochoa, J.L.: im4Things: an ontology-based natural language interface for controlling devices in the Internet of Things. In: Alor-Hernández, G., Valencia-García, R. (eds.) Current Trends on Knowledge-Based Systems, vol. 120, pp. 3–22. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-51905-0_1

14. Huang, J., Zhou, M., Yang, D.: Extracting chatbot knowledge from online discussion forums. In: IJCAI, vol. 7, pp. 423–428 (2007)
15. Desai, A., Oza, R., Sharma, P., Patel, B.: Hypervisor: a survey on concepts and taxonomy. Int. J. Innov. Technol. Exploring Eng. **2**(3), 222–225 (2013)
16. Morabito, R., Kjällman, J., Komu, M.: Hypervisors vs. lightweight virtualization: a performance comparison. In: 2015 IEEE International Conference on Cloud Engineering (IC2E), pp. 386–393 (2015)
17. Celesti, A., Mulfari, D., Fazio, M., Villari, M., Puliafito, A.: Exploring container virtualization in IoT clouds. In: 2016 IEEE International Conference on Smart Computing (SMARTCOMP), pp. 1–6 (2016)
18. Bernstein, D.: Containers and cloud: from LXC to docker to kubernetes. IEEE Cloud Comput. **3**, 81–84 (2014)
19. Madan, A., Dubey, S.K.: Usability evaluation methods: a literature review. Int. J. Eng. Sci. Technol. **4**(2), 590–599 (2012)