SN Computer Science (2021) 2:306
https://doi.org/10.1007/s42979-021-00696-4

ORIGINAL RESEARCH

Research on a Communication Platform Coordinating Web Services and Smart Speakers on the Application Layer

Takeshi Tsuchiya[1] · Ryuichi Mochizuki[1] · Hiroo Hirose[1] · Tetsuyasu Yamada[1] · Norinobu Imamura[2] · Naoki Yokouchi[2] · Quang Tran Minh[3]

Received: 12 March 2021 / Accepted: 10 May 2021 / Published online: 28 May 2021

Abstract
We propose an information platform for connecting smart speakers to Web services and acquiring data. The proposed platform is based on the fog computing model. It enables the smart speaker to connect to web services, control these services, and collaborate with other smart speakers on the application level by converting protocols in the node. In this research, we developed a service to retrieve news about artists of interest by utilizing the history of music services. Furthermore, we constructed the service connecting Spotify and Google News. Then, from the viewpoint of application development, we evaluated the practicality of the proposed platform and its development potential. We found that the unified communication platform reduced the logical source lines of code by about 35% and enhanced the ease of development and serviceability. In addition, We discussed the possibility of using the fog computing model to collaborate data among various http-enabled smart devices and web services as future work.

Keywords Smart device · Distributed platform · Web service · Cloud computing

This article is part of the topical collection "Future Data and Security Engineering 2020" guest edited by Tran Khanh Dang.

✉ Takeshi Tsuchiya
tsuchiya@rs.sus.ac.jp

Ryuichi Mochizuki
gh20515@rs.sus.ac.jp

Hiroo Hirose
hirose@rs.sus.ac.jp

Tetsuyasu Yamada
yamada@rs.sus.ac.jp

Norinobu Imamura
imamura@bip.co.jp

Naoki Yokouchi
yokouchi@bip.co.jp

Quang Tran Minh
quangtran@hcmut.edu.vn

1 Suwa University of Science, Chino, Japan

2 Bip Systems Inc., Tokyo, Japan

3 Ho Chi Minh City University of Technology, VNU-HCM, Ho Chi Minh City, Vietnam

Introduction

Advances in the Internet of Things (IoT) technologies have led to the development of several types of smart devices that have both conventional functions and Internet connectivity. Smart devices improve the usability of the conventional devices by adding new information services and utilizing sensors and information from the Internet. Various newer smart devices and services are expected to be developed in the future.

However, currently, the available services are limited by the manufacturer, operating system, system environment, etc. In most cases, to use a particular service, a device capable of using that service is selected. Furthermore, the number of smart devices capable of working with the web services used daily on PCs and smartphones is restricted, and their availability is limited. Therefore, the use of smart devices has not spread to the general public. However, there exist business models with the strategy of "enclosing the user" Amazon Alexa [1], Google Home [2], and LINE Clova are competing with each other in Japan in terms of service functions.

From the viewpoint of users, all smart speakers are designed to register speech when particular predefined keywords are pronounced by users. Then, the voice is converted to text using an individual cloud. A smart speaker recognizes a specific task based on the obtained textual information, such as providing weather information or the day's schedule. Generally, there is no large difference in services between two smart speakers, though they may differ in terms of quality. Although differences exist in terms of data format and detailed protocols of smart speakers, these differences are minor. They are similar in that the service developer develops the protocol and data format, called skills, according to the standards of each smart speaker. Therefore, the execution of an application in a smart speaker is performed with a common data flow.

Currently, it is necessary to develop the corresponding protocols and share the database for each skill to integrate the skills of smart speakers with general web services; that is development is required for each connecting web service, thereby limiting the number of skills available for data integration. At this point, if the above-mentioned smart speakers with similar data flow and http, which is the interface of web services, can be converted to each other, it is expected that these coordination services can be provided by defining the mapping between the existing skills and the protocol (http).

In this research, as a specific target for development, we consider a smart speaker. The proposed platform enables users to connect to web services through various devices, including PCs and smart phones, thereby overcoming the limitation of using a particular device only for a specific service.

In this paper, we propose a unified communication platform for smart devices having a similar data flow, which enables data integration with web services by absorbing the differences among data formats, protocols, and development environments. The proposed platform enables integrating smart devices with current web services using a method of mutual transformation with http. In comparison to the conventional smart devices that are independent for each service, the proposed platform enables smart devices to access the same data as the PCs and smartphones used in daily life for each service, and to coordinate with other smart devices. Consequently, we can expect improvement of the usability of the smart devices as well as the advancement of services via their integration with the existing web services.

In this paper, we first clarify the differences in skill development and data flow among typical smart speakers. Then, the functions required for unified communication platform are explained. As an application of web service data coordination, the development of a skill that uses the usage history of a music service to provide information on news related to an artist of interest is described. Furthermore, the feasibility of development of services coordinated among smart devices using the unified communication platform and the improvement in serviceability via coordinaton with web services was evaluated. The evaluation results are presented herein.

Related Research and Approaches

This section describes related works and introduces a novel approach on which the present work is based. Furthermore, the issues involved in applying the previous works to this research are described. Finally, an approach to solve these issues is presented.

Related Research

Fog computing was recently proposed to improve the throughput of cloud by distributing cloud functions to the edge nodes of the network [3, 4]; in this model, the fog nodes at the edge of a network providing the functionality of a conventional cloud network (e.g., data management and specific data processing functions).

A fog node works as a service gateway and a local server for users. It enables the transparent implementation of data processing in cloud networks and middleware. Load-balancing methods for cloud networks based on fog nodes have been widely discussed. We propose a method to ensure the privacy of the information managed in user groups, utilizing fog nodes as a private proxy function.

In this work, we assume that fog nodes are allocated to each data unit, i.e., the information managed by user groups. In [5], the aforementioned fog computing model was applied to personal data management. The speech data describing medical conditions that contained personal information was processed through speech analysis in a fog node. Then, only the text data corresponding to a diagnosis were sent to a cloud network. Fog nodes dynamically determined whether the information should have been sent to a cloud, and whether the privacy of information was protected appropriately. This method allows protecting the user's privacy and simultaneously reduces the voice-processing load in a cloud network.

In [6], a fog computing model was applied to a distributed IoT environment. However, this model was not feasible for integrating all information when sensors were distributed on a large scale because of large costs incurred to aggregate all sensed information. Therefore, information was aggregated and processed near a sensor and then aggregated and sent to a server. Here, fog nodes were utilized as servers aggregating various sensed information in a local network. Moreover, a method to synchronize the information from multiple fog nodes in a cloud network was proposed.

In the present research, we focused on processing textual information managed by a user group instead of the sensing information aggregated in a local server. For this purpose, we had to combine and synchronize the data from multiple fog nodes to apply a machine-learning analysis. Therefore, it was necessary to introduce a new method to link information among fog nodes, as the function of a specific cloud network is insufficient.

Approaches to Problems

The development environment on smart devices is more limited than those of conventional computers. Therefore, we adopted the fog computing model wherein fog nodes are deployed near a smart device [7].

A fog node behaves as a gateway and connects external web services to a smart device, as shown in Fig. 1. These nodes provide the functions that are necessary for web service clients, but are unavailable in smart devices. For example, the use of APIs provided by web services and the registration of user login information can be realized on a smart device by interconnecting skill (applications) of nodes with a smart device in advance. Considering the variety of types of smart devices, it is necessary to connect the standard web services using fog nodes. Thus, it is possible to share information with other smart devices, and an interconnection among different types of devices can be realized.

The smart devices targeted in this work are mainly watches and speakers. These devices have a restricted development environment and software execution capability compared to computers and smart phones in terms of operability and screen size. In this work, we adopted the fog computing model that implies deploying fog nodes near a smart device [8]. Therefore, we placed fog nodes near these smart devices, as shown in Fig. 1, to overcome the above-mentioned limitations of smart devices by converting protocols and data formats at these fog nodes.

To connect a smart device to the web, at least one interface should be available for sending and receiving data via http, entering user authentication information, and inputting to and outputting from the smart device. These functions should be complemented and supplemented by skills corresponding to the applications of the smart speaker and the functions of the fog node. Then, the smart devices must have a software platform corresponding to the skills (applications) and fog node functions in advance. The use of fog nodes enables these devices to access the web services that can be expected to enable information sharing and interconnection among smart devices where such coordination cannot be realized with each other conventionally.

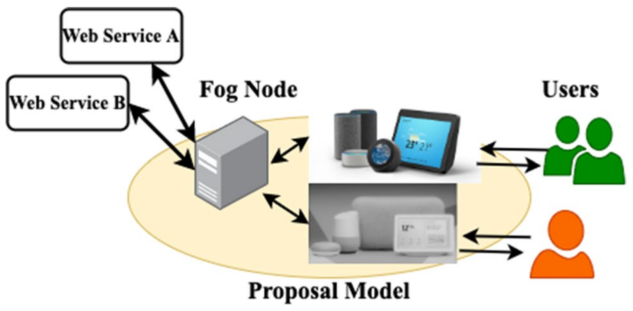

Fig. 1 Outline of the proposal model

Functions of a Unified Communication Platform

In this section, the skill development methods used to develop applications for Amazon Alexa and Google Home, which are representative smart devices are explained. Furthermore, we clarify the functions of the proposed unified communication platform for connecting to web services according to their commonality. Finally, the design of the proposed unified commucation platform is described in this section.

Skill Development in Amazon Alexa

For realizing a web service connecting to Amazon Alexa and an external web data source, the Amazon web service (AWS) by Amazon must be used in combination with a web server running an application. As shown in the upper part of Fig. 2, AWS employs Alexa to extract user intentions from voice information and utilizes Lambda to execute the program code based on the retrieved user intentions. This means that smart speaker applications (skills) are managed by Alexa and executed by AWS Lambda using the information acquired from user conversations. Utilizing https, this procedure can be executed on an arbitrary web server outside an AWS network.

Alexa skills can be created by defining several types of intents in the Alexa developer console [9]. Defining an intent means specifying the relationship between a text message that is acquired based on pronounced words and a task to be processed. An intent can be considered as an intention of a speaker. Each intent may also correspond to a variable called a slot (for e.g., the weather today). Intents can also contain variables such as date, time, and location set in the speaker, called slots, and these variables are inserted when the text is converted.

The skill to be executed on Lambda is defined in the JSON (JavaScript Object Notation) format shown in code 1. Similarly, the response is also predefined in this format, and information about the slot is inserted into the processing result as necessary. All of this information is communicated via http to ensure compatibility with web services and easy conversion in the fog node.

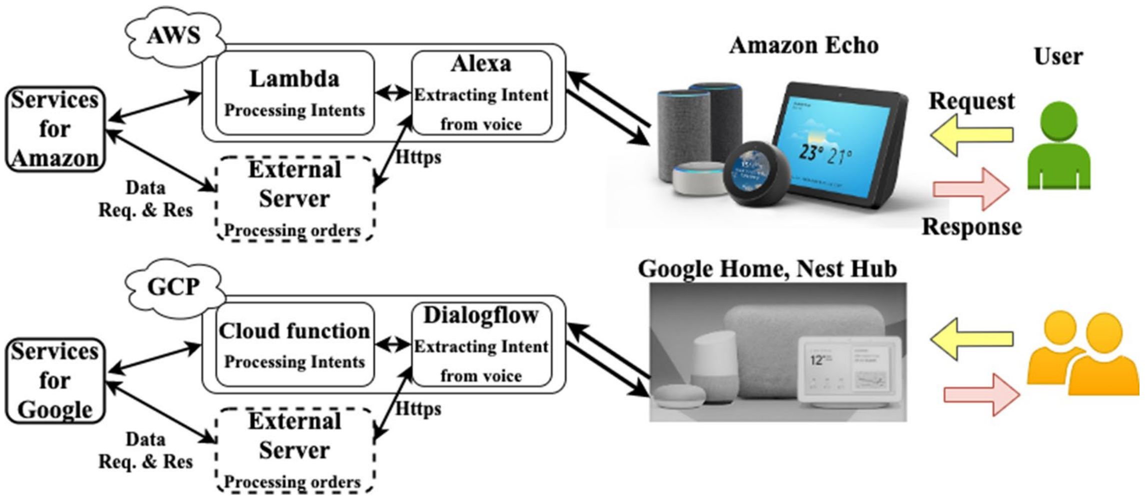

Fig. 2 Development environment of smart speakers

Code 1. Receiving an intent from AWS

```
{
    "version": "1.0",
    "request": {
        "type": "IntentRequest",
        "requestld": "amznl.echo-api.request.3180b2f-2980
        -406b-8cb4-921073729ddd",
        "timestamp": "2020-03-18100:57:221",
        "locale": "ja-JP",
        "intent": {
            "name": "ShowTomorrowTaskIntent",
            "confirmationStatus": "NONE",
            "slots": {

                "dateslot": {
                    "name": "dateslot",
                    "value": "2020-03-19",
                    "confirmationStatus": "NONE",
                    "source": "USER"
                }
            }
        }
    }
}
```

Skill Development in Google Home

Skill development in Google Home is shown in the lower part of Fig. 2. The process is generally similar to that of Alexa but requires using the Google Cloud Platform (GCP). Here, the user's speech is sent to Dialogflow on the GCP and executed in the Cloud function, which is the program execution environment. Here, similar to the case of Alexa, the program can be executed by an external server by connecting via https.

The skills executed in the Cloud function are defined in the JSON format shown in code 2. Responses are also predefined in this format, and the slot information is inserted into the processing result as necessary. Similarly, all of the communication is performed using http.

Code 2. Receiving an intent from GCP

```
{
  "responseId": "7c5a13ad-5201-4add-bf35-4dd8c9b4390e-dd2bbea9",
  "queryResult": {
     "queryText": "Let me know my task until tomorrow",
  "parameters": {
     "date": "2020-03-26T12:00:00+09:00"
     },
     "allRequiredParamsPresent": true,
     "outputContexts": [
        {
          "name": "projects/sample-todoapp",
          "parameters": {
           "no-input": 0,
           "no-match": 0,
           "date": "2020-03-26T12:00:00+09:00",
                 "date.original": "Tomorrow"
           }
        }
     ],
     "intent": {

          "displayName": "showTaskIntent"
       }
    }
}
```

The unified communication platform receives user intents in a textual format from each environment and executes them by itself instead of using a cloud. Accordingly, the proposal platform can be used as an interface for external services to acquire and control data, as shown in Fig. 3. A response from an application is received by such an integrated platform. Then, the response is adapted to the requesting environment of a smart speaker and is sent to a user in a form of a final response.

Overview of the Unified Communication Platform

In developing the smart speaker skills as described in the previous subsection, the data flow of extracting the speaker's intent, executing the intent, and responding to the user with the processed results remains the same. However, the execution environment, the actual protocol to be coded, and the data format of the parameters are different. Consequently, the proposed unified communication platform must define a protocol conversion for each smart speaker for connecting to the web service, though the execution environment, actual protocol to be coded, and data format of the parameters are different. Hence, the proposed platform must define a protocol conversion for each smart speaker for connecting to the web service. More specifically, each type of smart speaker should define the http connection request corresponding to the intent for each speaker and transformation of the response to this request into a response to the intent sender.

Conventional PCs and smartphones can display the result in a web screen by sending a connection request by http and

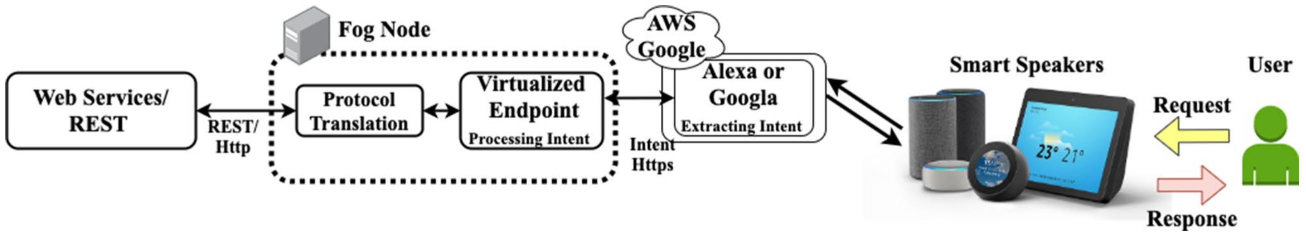

Fig. 3 Sequence of converting data in the proposed platform

receiving the response from the browser. However, for smart speakers, which are based on voice input, such visual display to instruct the user to make choices is generally not possible. Many smart speakers do not have a screen, and therefore, they cannot display responses from the web service. Thus, the unified communication platform provides connections to web services as user intents, including certain data operations and processing such as the type of data and the date of creation. The data handling available in the web service is not unified and is different for each connecting web service. Thus, data operations and processing are defined in advance for each web service to enable operation and processing of advanced data using intents. An increasing number of web services being operated through web interfaces such as REpresentational State Transfer (REST). Our proposed platform enables data operation and processing by defining the correspondence between the intent acquired from the smart speaker and the web interface of each web service.

The functions of above-mentioned platform are not available as middleware for smart speakers owing to limitations in the development i.g.available APIs, types of data, etc. Thus, the fog node is deployed between the client and the web service, as shown in Fig. 1, and is used to connect to the web service by executing transformations. The user deploys skills corresponding to the unified communication platform and plugins for the smart speaker. At this point, the connection destination of the skill of each smart speaker and the connection source to the web service are handled as fog nodes at the session level. That is, the fog node serves as an application gateway, an endpoint for the smart speaker, and a client for the web service.

Interface of the Unified Communication Platform

The interface to the smart speaker provided by the fog node is named "basement". The intent received from the smart speaker is converted to http for connection to the web service. Furthermore, the content is converted into operations adapted to the interface of the web service to be connected. Since it is not feasible to cover all the interfaces of individual web services in the basement, the respective interfaces must be added as plugins, as shown in Fig. 4.

Assuming that fog nodes are used to connect to several web services, the basement must manage several plugins concurrently. The basement must determine the plugin that is used by the intent received from the user; in other words, the basement routes the intent to each plugin. To achieve this purpose, the name of the web service to be connected is assigned when registering the plugin, and each operation interface is registered in the basement with the plugin. When an intent is received, it will be routed based on this information.

Most of the web services to be connected require user authentication. Current smart speakers do not have a way to manage user information. Hence, it is necessary to store user authentication information for each web service. Thus, the basement must have a database function to manage the intent names and user authentication information related to the aforementioned web service plugins. An ID is assigned to each information managed for identification. This means that the fog node manages the routing and web service connection information based on the IDs of the intents it processes.

Currently, smart speakers can only identify the user by the service account registered on the speaker itself. In this case, the same type of smart speakers is identified, but a different type of smart speaker cannot be identified. The unified communication platform allows the fog node to identify the

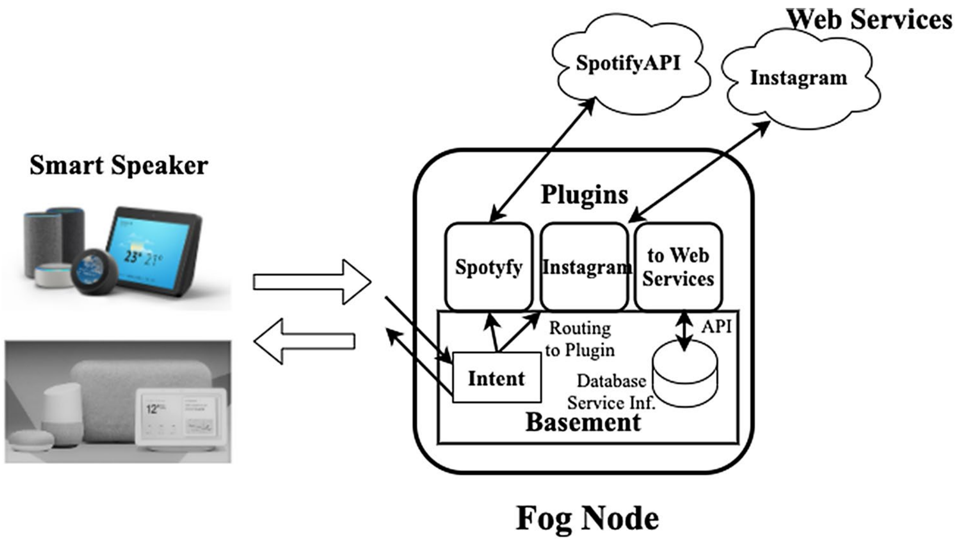

Fig. 4 Design of the unified communication platform

user using a combination of the account and the ID of the intent. In addition, the same plugin can be identified in web services by different authenticated users. Hence, a fog node can be used for smart speakers of multiple users.

Interface for Web Services

From the information contained in the intent, the ID is used for routing to the plugin. Then, the plugin makes a connection to the web service based on the interface defined in the plugin. The intent is converted into a REST or similar web interface exposed to the web service. Corresponding to the interface exposed by the web service, it is necessary to correspond the conversion between the intent and the interface function for each kind of smart speaker for each web service. The definition of this mapping is called a plugin in this paper.

Responding to Smart Speakers

Responses acquired from web services by the execution of intents are assumed to be diverse; they include textual information responses, text web pages containing a large amount of page links such as search results, and receiving streaming data such as music services as well as no responses. The conventional smart speakers focus on no response items and listenable data such as text information readable text information, and voice data. Currently, smart speakers with displays are being increasingly used, and the number of set-top-box type devices used with TVs. In such a scenario, the unified communication platform should be able to display responses. Furthermore, a method to display the response as a dynamically generated web page is considered for a large amount of text information, and a method to display the contents as an image is investigated if page generation is not possible.

This function will be realized as one of the functions of the basement, as it responds to the smart speaker.

Table 1 Environment of implemented proposal platform

Function	Environment
Dev language	Node JS v14.10.0
Library	Google Assistant Conversation 3.1.0
Database	NeDB 1.8.0

Implementation

This section describes the implementation of the unified communication platform. After implementation, the platform connects to web services based on the intent sent from the smart speaker. The implementation is based on the environment presented in Table 1.

Implementation of Basement

The basement is implemented as the basement class for managing routing and authentication information for web services.

The basement identifies the application based on the value of the "IntentRequest" field in Alexa and the "intent" field in Google from the intents received in the form of code 1 and code 2. Then, these intents are routed to the corresponding plugin. Connecting to a new web service is made possible by storing the application interface registration and plugins in the basement (Fig. 4). The data formats of fields listed in Table 2 are specified, and the same ID is added to the intent containing this value. The parameters of the intent are converted to the http format as the request body to the web service. Subsequently, the intent received from the smart speaker is generated as an event, and it enables to access to the application via the plugin.

The user authentication information for the web services managed in the database shown in Fig. 5 is managed in the database for each ID listed in Table 2. The user authentication information for Web services managed in the database shown in Fig. 5 is managed in the database for each ID shown in Table 2. In this case, the user authentication information includes not only the user and password, but also the access token for accessing the service via web API. Moreover, smart Speaker itself cannot store data; hence, the data acquired from the plugin are managed in the database. This enables coordination among web services using the acquired data.

Currently, our implementation of the response to the user of smart speakers is completed up to a point where it can be displayed as a dynamic web by converting the response into XML, besides reading out the conventional text information. At present, the response display function using images is being developed as a solution for smart speakers that cannot display web-based information. Conventional smart speakers are based on the assumption of synchronized communication originating from the user. However, some applications

Table 2 Managed information on the basement DB

UUID	Identified name	JSON
Unified ID number	Intent name (web service name)	user: user1, password: PW, token: xx5xxAx

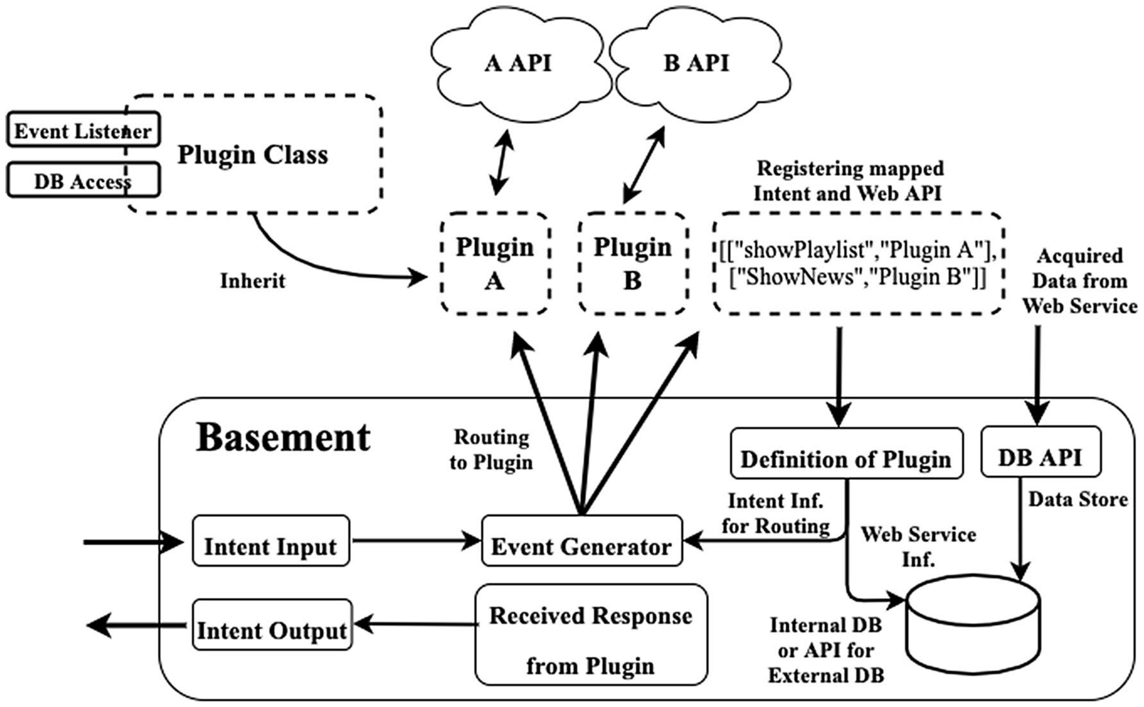

Fig. 5 Implementation of the unified communication platform

may require sending push notifications to the user; hence, the implementation of the basement includes simulated push notifications from the application using periodic requests. An empty intent is periodically sent and received between the skill and basement to enable notification from the application.

Implementation of Plugins

Plugins are implemented individually according to the interface of the web service to be connected. Adding a new plugin necessitates the development of the items indicated by the wavy lines shown in Fig. 5. Plugins are described in the control functions of the web interface, and their information is registered in the basement for routing. The current implementation of plugins is created by inheriting from the plugin class, thereby enabling coordination with the basement. The intent received in the basement is routed to the plugin as an event. The plugin watches these events, and if it detects any, it acquires the information necessary to connect to the web service (user, PW, token, parameters, etc.) from the database using the basement API. The plugin then sends an http request to the corresponding web service. Then, the response from the web service is converted at the fog node to suit the smart speaker. If the response is written in json data, it is converted into text information that can be spoken by the smart speakers. Conversion to the data type suitable for the smart speaker is executed in the basement, while the plugin handles the data processing of the response.

We implemented plugins for the web services Spotify [10], Instagram [11], and Google News [12]. As an example of the Spotify plugin, the intent was mapped to the web API (see Table 3). The current implementation supports acquisition of its own playlist and playing songs.

Evaluation

To evaluate the proposed unified communication platform, we constructed a service for acquiring data from Web services using the implemented plugin and collaborative services using this data.

Overview of the Web Service Collaboration Service

A service using the Spotify and Google News plugins (Sect. 4.2) was constructed. These implemented plugins enable connecting smart speakers to each web service. The

Table 3 Mapping to Spotify

Function	Intent	Web API
Show PlayList	showPlayList	GET /v1/me/playlist
Play music	Play	PUT /v1/me/player/play

advancement of the service using these plugins is discussed herein.

The current implementation of Spotify enables acquisition of its own playlists and playing music, and that of Google News enables search and acquisition of the latest news collected by keywords from web articles. By combining the data provided by these web services, the most recently listened artists from Spotify playlists and the most frequently listened artists from the last 50 plays are identified. Then, the service retrieves and displays the news related to these artists.

This means that the users can get news regarding their favorite artists in priority. In our collaborative service, Spotify playlists are stored in a database provided by basement in advance, and the number of times they are played is counted to classify the most recent artists of interest to the user.

Scenario for Evaluation

The constructed collaborative service was evaluated by comparing the workloads of the source code between scenarios with and without the proposed platform. The source code was compared using the logical Source Lines of Code (SLOC).

Table 4 Results of logical source lines of code (SLOC)

Method	Number of SLOC
Use of proposal unified communication platform	107 steps
Full scratch buildings	165 steps

Results and Discussion

The results of the evaluation are listed in Table 4. Using the proposed unified communication platform, the logical SLOC can be reduced by about 35%. In both cases, it is necessary to define the web interface as a plugin for each connecting web service. With the unified communication platform, the connection to a new web service requires the registration of the interface operation name of the plugin for the placement and routing of the plugin, as shown by the wavy line in Fig. 5. On the other hand, in the collaborative service (Fig. 6), which is built from scratch in the same way, the correspondence between the intent and each interface included in the plugin is required for each newly connected web service. The database settings in the fog node are also set for each new one. This means that the use of the unified communication platform has reduced most of the functions provided by the basement. Therefore, the use of the proposed the unified communication platform reduces the logical SLOC caused by processing of low-layer functions in the platform.

The reduction of the logical SLOC does not indicate directly the significance of the proposed method. However, it indicates that the use of the unified communication platform makes it possible to connect to a variety of web services with little development, while the current smart speaker is restricted to the use of specific services corresponding to its skills. The evaluation results also show that development efficiency can be improved by about 35% even for a simple collaborative application. The results indicate that the unified communication platform improves the development of smart speakers and enhances their availability by expanding their range of applicability. Moreover, the system realized data sharing among devices that are not directly compatible

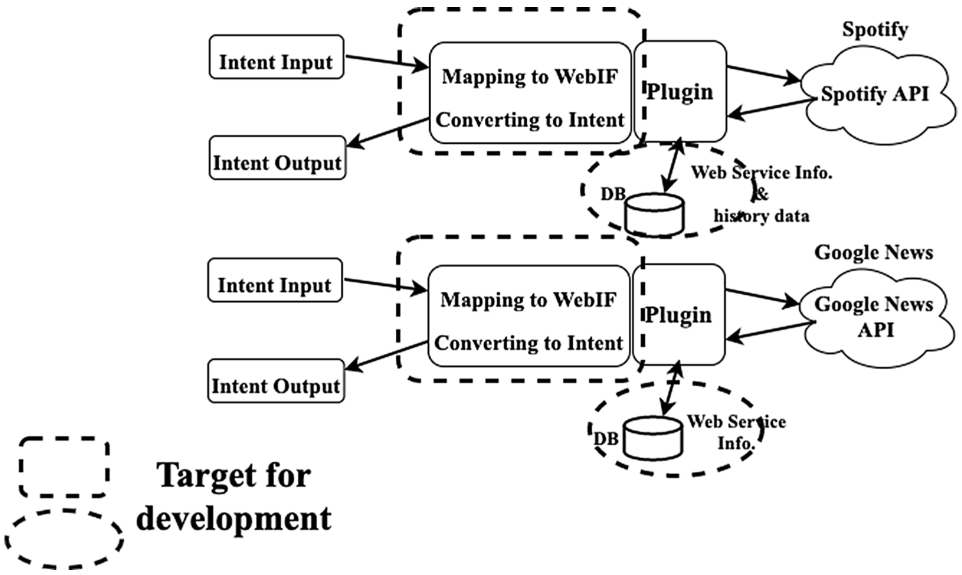

Fig. 6 Implementation of full scratch buildings

using the general http used in PCs and smartphones and converting data in fog nodes. Hence, it is expected that the proposed platform can contribute to the creation of new services in the future.

Future Issues

Connecting to new web services requires the development of plugins and registration with the basement. The future issue for the unified communication platforms is not writing program codes in development, but registering intents and mapping them to web interfaces are relatively easy. Therefore, a method for developing fog nodes externally using the web and automating the development process through patterned development can be considered in the future.

The proposed method is based on the idea of improving the development of data collaboration services among web services as an information infrastructure for smart devices by adding not only the current protocol conversion and database functions but also the functions required for data collaboration.

For displaying the responses from web services that cannot be read out, we are considering improving the usability by integrating services of any type, such as by attaching an image of the response to an e-mail or displaying it on the user's smartphone.

Conclusion

A unified communication platform for connecting all types of smart speakers to web services is proposed. This platform enables various web services and devices to communicate with each other via the connection to web services. We implemented a plugin to connect to Spotify, Instagram, and Google News, and constructed the service connecting Spotify and Google News.

Our evaluation showed that the proposed platform reduces the logical SLOC by about 35%, and improved the ease of development and serviceability. It is expected that the platform can be made compatible with many data types' conversion via http. In the future, we would like to consider methods to improve the ease of development and usability by providing functions as an information platform for smart devices.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Amazon Alexa. https://alexa.amazon.com/.
2. Google Home. https://developers.google.com/assistant.
3. Bonomi F, Milito R, Natarajan P, Zhu J. Fog computing: a platform for Internet of Things and analytics. In: Big data and internet of things: a roadmap for smart environments. Cham: Springer; 2014. p. 169–186.
4. Alrawais A, Alhothaily A, Hu C, Cheng X. Fog computing for the Internet of Things: security and privacy issues. IEEE Internet Comput. 2017;21(2):34–42.
5. Dubey H, et al. Handbook of large-scale distributed computing in smart healthcare; 2007. p. 281–321.
6. Abuseta Y. A fog computing based architecture for IoT services and applications development. Int J Comput Trends Technol (IJCTT). 2019;67(10).
7. Tsuchiya T, Mochizuki R, Hirose H, Yamada T, Koyanagi K, Minh QT, Dang TK. Dynamic data management strategy on cloud network by fog computing model. In: Int'l Conf. on Future Data and Security Engineering (FDSE 2018), Springer Lecture Notes in Computer Science, vol. 11814, Vietnam; 2019. p. 332–342.
8. Minh QT, Huu PN, Tsuchiya T, Toulouse M. Openness in fog computing for the Internet of Things. In: Int'l Conf. on Future Data and Security Engineering (FDSE 2018), Springer Lecture Notes in Computer Science, vol. 11814, Vietnam; 2019. p. 343–357.
9. Alexa Developer Console. https://developer.amazon.com/ja-JP/alexa.
10. Spotify. https://www.spotify.com/.
11. Instagram. https://www.instagram.com/.
12. Google News. https://news.google.com/topstories.