A New Approach to Managing Smart Environment Infrastructure Settings

Gennady Dik[1,2(✉)], Alexander Bogdanov[1], Nadezhda Shchegoleva[1], Aleksandr Dik[1], and Jasur Kiyamov[1]

[1] St. Petersburg University, Saint Petersburg, Russia
{a.v.bogdanov,n.shchegoleva,a.dik,z.kiyamov}@spbu.ru
[2] St. Petersburg LLC "System Technologies", Petersburg, Russia
g.dick@systechnologies.ru

Abstract. The main problems of constructing and configuring of automation systems based on the "Smart Environment" technology come from the need of combination of heterogeneous software and hardware from various manufacturers in a single information space. The article focuses on the speech recognition method of setting up and training of the "Smart Ecosystem" using innovative artificial intelligence (AI) technology.

Keywords: Smart House · Smart Ecosystem · Smart Habitat · specialized software platform · natural user interface · speech recognition

1 Introduction

One of the significant factors of improving the quality of life of the population in modern conditions is the use of so-called "Smart Environment" technology, which, in the course of development, turns into a whole strategic direction of building habitat infrastructure called "Smart Habitat" (SH) or "Smart Ecosystem", consisting of elements of a "Smart System", IoT, etc. (Fig. 1) [1].

These elements of the "Smart System" can interact with each other by connecting via an internal network or switched via the global Internet. At the same time, there is increasingly a need to integrate not only different types of Smart System elements, but also those developed by different manufacturers, for example, smart lamps from Xiaomi and Philips HUE. In addition, owners (customers) of such installations increasingly have a desire, and sometimes a need, to manage them remotely using a computer, mobile device or voice assistant.

Recently, the most popular option is the ability to voice control not only individual devices, but also individual scenarios and extended scenarios as part of the "Smart Environment". This area has received especially active development with the appearance on the modern market of a large number of smart assistants such as "Alice" (a Yandex service), "Siri" (an assistant from Apple), "Google Now" (an assistant from Google), "Marusya" "(VK service)" "Salyut" (Sbera service), Alexa (Amazon) and others [2].

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
O. Gervasi et al. (Eds.): ICCSA 2024 Workshops, LNCS 14821, pp. 405–420, 2024.
https://doi.org/10.1007/978-3-031-65308-7_28

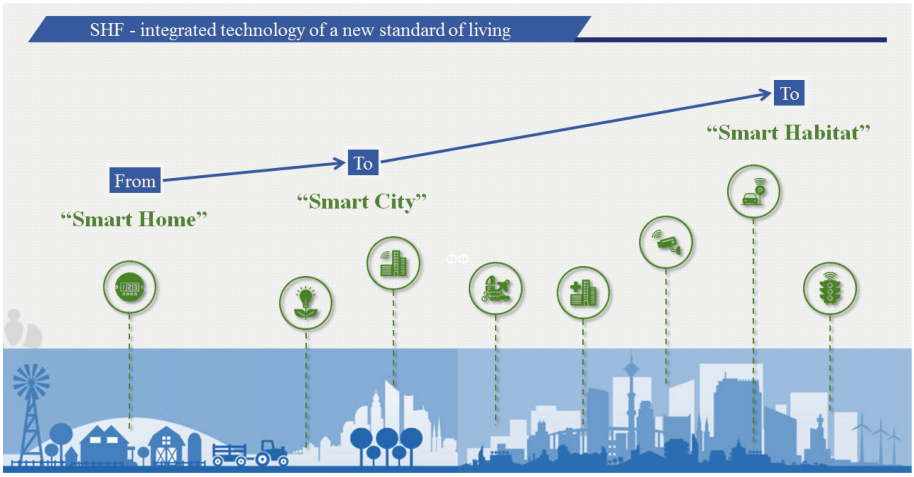

Fig. 1. SH development

The main prerequisite for voice control was the use of a user-familiar method of interaction between a person and a computer, implemented using a natural "communication" interface, which implies the use of innate means by a person such as gestures, voice, gaze. It should be noted that the most natural means of interaction for a person is speech, therefore the speech user interface has a number of advantages: minimal training time for the user to use the system, simplicity and high speed of generating requests to the system. This was the reason that made it possible to actively use the speech user interface in Smart Systems to control devices and scenarios.

And if the issue of managing fragments of SH ("Smart Home", "Smart Building", "Smart Office", "Smart Enterprise", etc.) is gradually finding its niches thanks to various innovative solutions, then the issue of customization and further training or reconfiguration (modernization) "Smart systems" is still a kind of "magic" of a narrow circle of specialists. Only these specially trained employees can make the necessary changes from the simplest to the most complex reconfiguration in the work of the SH, while spending serious material and time resources, including the Customer, not to mention various inconveniences and additional expenses.

The approach outlined later in the article proposes not only a new approach to the issue of setting up Smart systems, but also using the capabilities of AI for innovative training and subsequent equipment modernization in order to increase the efficiency of managing intelligent components of not only the "Smart Home", "Smart Building" etc., but also SH. At the same time, the novelty of the proposed approach consists primarily in carrying out voice tuning of the training system, as well as the possibility of subsequent modernization of the control system of the fragments of the SH.

2 Analysis of the Problems of Installation and Modernization of "Smart Systems"

"Smart Home", as a separate fragment of the "Smart Ecosystem", is a fully automated set of equipment that allows you to manage various household processes. The apartment owner controls the equipment using a remote control, an application on a mobile phone or using voice commands. The system may include household appliances, video and audio equipment, devices for controlling the microclimate in the room, etc. A necessary option is also control over the security system: surveillance cameras and alarms. In addition, a "Smart Building" system with additional capabilities is installed in private homes: control of water supply, heating, gas supply, etc.

And if all systematic steps for planning, design, installation and development of "Smart Systems" are defined and relatively unified, then most of the issues related to configuration, expansion of functionality, training and development for further operation remain open [3]. The combination of these issues develops into a serious problem for the following reasons:

1. Lack of a generally accepted concept for using the "Smart System" at the initial stage of its installation. The Customer (User), despite the great desire to explore all the possibilities and methods of using "Smart Things", various IoT, etc. in his environment usually does not fully understand the practical orientation of the "Smart Environment". He needs time both to get used to it and to acquire the necessary skills for further and future use.
2. Constant updating and expansion of the fleet of sensors and devices (capabilities of the "Smart System") and, accordingly, changes in the software of "Smart Systems" leads to an increase in the number of installed sensors and devices (capabilities of the "Smart System") about which the Customer (User) is often not aware This circumstance necessitates modification (if we are not talking about some serious reconfigurations) of the software and hardware and customizing it to your requirements.
3. Reluctance or impossibility due to temporary or financial difficulties to call specialists with sufficient professional skills to carry out highly qualified tuning work. Obviously, employees of Smart Systems companies will not refuse the request, but their work may be carried out at an inconvenient time for the customer, postponed due to workload for a long time, etc. What if you need to perform some simple setup operation, etc.?
4. The presence of heterogeneous systems from different manufacturers, which causes great inconvenience in managing these devices and systems as part of a single information space, not to mention the possibility of their uniform configuration or training.
5. The possibility of leaving (at best, temporary) a particular brand for technical reasons or other reasons, as well as ceasing support for outdated equipment. A striking example of this fact is the inability to connect Redmond's "Smart Appliances" to the Sky smart platform in November 2023. Despite the fact

that the reason was the debt of suppliers who had not paid for the corresponding licenses, customers were faced with malfunctions of the purchased devices. This problem was solved only in 2024. At the same time, the developers of the "Smart Home" components "Insteon" simply turned off their servers and the equipment lost its smart functions, and then stopped responding to control commands. In December 2023, the Yandex Smart Home did not work for about a day due to an accident on the Alice servers. The list of similar situations can be continued, but you should understand that if, due to a failure on the server, the light in the toilet stops turning on, this is not a big problem. But if, due to a sudden shutdown, the alarm signal from the security or smoke sensor is not received, the situation could turn into a disaster [4].

And this is not a complete list of problematic issues related to setting up and training Smart Systems. In order for the slogan "Welcome to the world of "Smart Systems", where technologies make our lives more comfortable and convenient" not to sound so openly provocative or frightening, but to demonstrate a real approach to the operation of the "Smart Environment"", this article will present results of a study to create a unified platform with the ability to integrate a training system based on an intuitive natural interface - a speech method of entering information.

3 Development of a Specialized Software Platform

To create our own prototype of a specialized software platform (SSP), it was carried out in 2 stages:

Stage 1 - research and analysis of methods for constructing an SSP that combines fragments of heterogeneous "Smart Systems" from various manufacturing companies in a single information space.

Stage 2 - analysis of alternative ways of teaching "Smart Systems", taking into account the features of speech input.

At stage 1, the main attention was paid to the problems of functioning of the "Smart System" on a cloud-based and local (on-premise) resource. How does the local version of the Smart Environment differ from the cloud version? Systems from Yandex, Google, Apple and Xiaomi, etc. rely on cloud services. And they, as it turns out, can easily disconnect users based on geography, be attacked by hackers, or stop working for some other reason.

Therefore, to build an SSP, the option of building a "Smart Environment" on a local resource was proposed. The "on-premise" solution has a number of significant advantages compared to the "cloud-based" option. An important advantage is the ability to control the information flow circulating in the SSP, because information is stored on a local server (within a controlled area), and, therefore, the access control table, audit log and other security requirements are controlled, while cloud solutions cannot guarantee the safety of information due to the risks of DDos attacks, etc. The "on-premise" option has advanced infrastructure settings for the Customer's needs (flexibility), the ability to operate in real time

(a sensitive parameter for SSP) and compliance with information security requirements (anonymization of personal data, compliance with requirements for the safety of corporate and other information). At the same time, systems in the "on-premise" version can receive commands from the User via the Internet, but all devices interact with each other in the "Smart Environment" via a local network.

The characteristics of the local server must be no lower than:

- CPU: Core i5-13600KF or Ryzen 5 7600X.
- RAM: minimum 8 Gb DDR4, Memory frequency: 2666M or more powerful.
- HDD for datasets: 1 TB or more.
- SSD: at least 128 GB.

The configuration is determined based on the need to support speech input.

There are several popular ready-made open source home automation platforms. Almost all of them allow you to configure devices and Smart Environment scenarios through user-friendly applications or a web interface. The list of the most common systems should include:

- "Home Assistant" is the most popular system, it is easiest to learn with instructions.
- "OpenHAB" is a somewhat "overcomplicated" system for creating scripts; programming skills are required.
- "Domoticz" is a fairly functional system, but the interface looks outdated.
- "ioBroker" - slightly less compatible with various systems than Home Assistant.
- "Node-RED" is a difficult system to master, as it is based on visual programming.
- "Apple HomeKit" is a home automation system where control centers are produced directly by Apple, and other devices can be produced by third-party manufacturers, but must undergo mandatory certification. Along with high quality and security, and the use of Siri for voice control, the main disadvantages include cross-platform issues and extremely high cost.
- "Samsung HomeKit" includes everything you need - from control centers to sockets, switches and light bulbs with the ability to connect devices from other manufacturers through the SmartThings gateway, using voice control with Google Assistant and Amazon Alexa. But at the same time, the necessary set of components for the minimum version of a "Smart Home" is practically absent.

Among the reviewed and tested systems, the "Home Assistant" (HA) is of greatest interest as a prototype of the proposed SPP, as a system that uses open source code as software (Fig. 2).

In addition, the feasibility of using this system follows from its following capabilities:

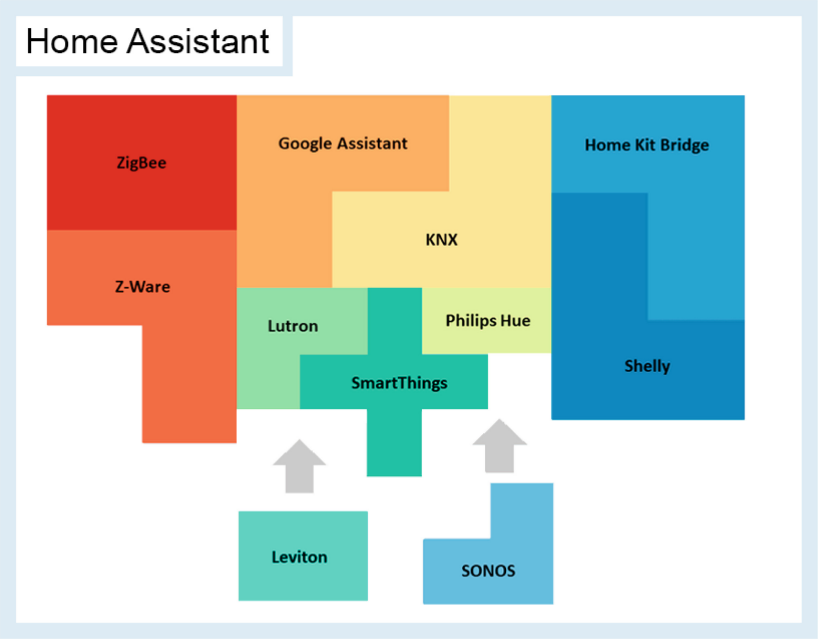

Fig. 2. Integration of heterogeneous systems into the Home Assistant platform

1. Support for devices from different manufacturers.
2. Implementation of complex automation scenarios with the ability to use voice control (assistants) and visualization via a web interface.
3. Setting up applications for managing "Smart Systems" via mobile devices and other types of IoT.
4. Supports major communication standards, including widely used Wi-Fi, Bluetooth, ZigBee and Z-Wave protocols.
5. Integration of security and fire alarm systems, access control systems (intercom communication systems) and video surveillance to organize a complex of technical security systems.
6. Using SQLite as a database with an interface based on SQLAlchemy and, as a result, supporting a fairly wide range of relational DBMSs, namely Microsoft SQL Server, MySQL, MariaDB, PostgreSQL and others. YAML syntax is used for configurations and settings. Although most integrations are configured through the user interface, some tasks require editing the main configuration file

At the same time, "Home Assistant" supports various installation options:

1. HA Operating System (recommended by the developer): installs the program as an operating program for working in home automation tasks, includes the kernel, the Supervisor function for managing the kernel, a set of pre-installed plugins for integrating devices and accessing services.

2. HA Core: manual installation of the kernel using the Python virtual environment (suitable for experienced users), gives full access to the platform, requires minimal knowledge of the operating system and the ability to work with the Python environment and the command line.
3. HA Operating System is HA Supervised: installing the platform kernel with the Supervisor function in manual mode (suitable for advanced users), provides the most complete control over the system, but in this case only works with Linux.
4. HA Container: installing the kernel in Docker containers.

The HA software platform includes integration for connecting Google Assistant and Amazon Alexa, which makes the use of voice assistants possible. A problem for the user may be the monolingual interface (English only) for the Amazon assistant and not always correct processing of information from Google devices. In this case, in the context of building a "Smart Ecosystem", it is advisable to use a version of the HA platform with manual installation of the kernel, which will allow creating the software basis for the SPP prototype. Taking into account the open source code of the HA platform, the current question is to develop language service code in accordance with the documentation on the language API (Application programming interface) for the operation of voice assistants when interacting with the platform. The solution being developed is necessary to create a fully functional SPP with the subsequent integration (stage 2) of the capabilities of speech customization and training of the "Smart Environment" into it.

4 Justification of the Speech Method of Entering Information for the "Smart Environment"

Studies of various options for entering information into electronic devices and systems (text and voice input were compared) have shown that for a person, voice or speech input is a much more effective input method than typing on a keyboard. Therefore, it is not surprising that humanity has long been interested in the issues of interaction with computing and other intelligent electronic devices through speech.

The theoretical foundations for constructing a speech recognition system (SRS) are reflected in the classic works of L.E. Baum and his colleagues [5,6], where it appeared as part of solving the problem of creating a signal model, as well as extracting information about the source. Later, the idea of using the mathematical basis of Markov chains for SRS was presented in the works on Hidden Markov Models (HMM) by L.R. Rabiner and E.I. Bovbel [7,8]. It should be noted that the HMM theory was chosen as the methodological basis for creating the SPP module, but its presentation of its provisions is beyond the scope of this study.

Modern trends in the development of IoT have demonstrated that the future belongs to a comprehensive user interface that allows using, in addition to vision,

various human senses, primarily hearing and speech. These assumptions are based on the following facts:

- speech is the most popular form of communication in the human environment;
- hands and eyes remain free, which makes the voice interface a priority;
- there is no need for direct contact during interaction, since the microphone and speakers can be located at a distance;
- modern software and hardware have achieved a fairly high level of accuracy in speech recognition.

Research has shown that when replacing the input method from classic to voice, information will be better processed by the user as time is saved when typing and further transmitting information in the form of text to its destination. At the same time, in both cases there are technical errors, but with voice input there are errors associated with incorrect recognition (about 5–7%), which often leads to an increase in input time (additional input of words, phrases, commands). In addition, as practice shows, voice input becomes the most effective solution when the query length is 2 words or more, but this usually requires a keyword [9].

According to statistical data, the promise of speech input is due to the following:

1. Voice queries are 3.7 times faster than traditional ones.
2. Quick input is usually equivalent to a quick answer, and users prefer voice search for the sake of speed.
3. Voice search is widely used in IoT. According to Stone Temple, about 60% of users searching on mobile devices use Voice Search.

Taking into account modern trends, the applied use of voice information recognition systems looks quite logical and relevant for organizing information input for setting up and training the “Smart Ecosystem”.

In terms of application, for integration into SSP it is planned to use an already developed and tested tool, the so-called speech recognition “engine” or Automatic Speech Recognition (ASR) with an available API for embedding ASR into software applications.

In modern conditions, ASR is increasingly used in various fields of development of science and technology, namely:

- Transcription and dictation. ASR is widely used for transcribing audio recordings when generating various types of reports, saving lecture material, etc. These operations allow you to quickly and accurately convert voice into text form to further simplify the process of data processing and analysis.
- Speech recognition in automated and guided vehicles, which allows drivers (operators) to control various functionality of these vehicles.
- Voice assistants and control systems of modern “Smart Ecosystems”. Here ASR is a key component of voice assistants “Siri”, “Alice” and “Google Assistant”, etc. where users interact with different types of IoT, ask questions, and control devices using only speech input.

Thus, the above analysis showed the possibility of using SRS as an alternative way of entering information into the "Smart Ecosystem" for setting up and further training its individual elements and the entire system as a whole. At the same time, a number of limitations and assumptions related to the peculiarities of the implementation of the SSP should be taken into account.

5 Application of Speech Methods of Teaching "Smart Systems"

It is worth noting that the speech control of the "Smart Ecosystem" has been successfully implemented both on the basis of individual "Smart Systems" such as "HDL", "Xiaomi", "Philips HUE", "Google", "Yandex" and others, and in the combined environment like "Home Assistant" [10].

A study of existing algorithms for setting up and further modernizing the settings (training) of the Smart Ecosystem showed that the set of control instructions and the dictionary of words used are deterministic.

Standard similar set includes:

1. A package of instructions, which includes commands to control the activation of Smart Ecosystem devices depending on user actions or logical conditions (time, lighting, temperature, etc.). For example, "Enable", "Disable", "Increase", "Decrease", "Close", "Open", etc.
2. A list of devices, which includes elements of the "Smart Ecosystem" that are subject to the instructions. For example, "Lamp", "Sensor", "Warm floor", "Radiator", "Air conditioner", etc.
3. A list of locations, which includes definitions of places where this activation must be implemented. For example, "Kitchen", "Living room", "Dining room", "Toilet", etc.

Thus, the complete composition of the phrase for setting (training) may look like this: "Turn on the lamp in the kitchen", while the instructions for issuing for execution or reconfiguration may include other conditions, but for the correct functioning of the system, all of them must be determined in advance.

The process of preparing for operation using SRS can be implemented on the basis of a chatbot, which, upon command, asks questions like "What to do and with what? Where to do it? or similar. After completing the input, this information is visualized and presented to the Customer (user) in a graphical interface for approval of the final version of the Smart Ecosystem management. This method of entering setup and training information is strictly defined. At the same time, it should be noted that currently the following innovations have appeared for voice input - queries have become longer, and the style of queries has become more conversational, there is more human speech in them. In this case, an alternative way of entering voice training information in free form is possible, which significantly affects the input process and the operation of SRS. Therefore, the mandatory presence of keywords will allow us to form the correct

basis for setting up the "Smart Ecosystem". In this case, the SRS software must analyze the "key" phrases to save and prepare for approval by the user.

In the process of developing the SRS, it is necessary to take into account protection from incorrect recognition and extraneous noise, "cut-off" of the beginning and end of the input, the mandatory use of keywords, the possibility of correction and return to repeated input, etc. Thus, the procedure for recognizing and forming a database (DB) of training commands should be as follows:

Step 1 - receiving sounds into the recognition software application;

Step 2 - processing the received audio signal, assessing the quality and eliminating interference;

Step 3 - highlighting signal sections containing speech training commands using keywords;

Step 4 - speech recognition in selected areas, followed by saving in the database;

Step 5 - testing and adjusting, if necessary, the training database.

According to the requirement for the need for the functioning of the SRS in a local version (without an Internet connection), it is proposed to consider the option of implementing speech input of educational information in the Python software environment (hereinafter referred to as Python).

The following libraries can be used for speech recognition [11]:

1. SpeechRecognition - provides support for multiple engines and APIs such as Google Web Speech API, Microsoft Bing Voice Recognition and IBM Speech to Text.
2. PyAudio - designed for audio input and output in Python, provides the ability to use a cross-platform audio I/O library.
3. DeepSpeech - is an open source voice recognition software system based on deep learning.

A number of research experiments were conducted to develop recommendations for the use of open source speech recognition systems.

To this end, five open source systems are considered: HTC, Kaldi, Julius, and RWTH ASR. The choice of these systems is based on the fact that they are currently the most frequently mentioned in modern research papers and are used by software developers [12–15]. The selected systems were compared in terms of such indicators as accuracy and speed of recognition, ease of use and internal structure.

The accuracy of the system was compared using the most common metrics [16]: Word Recognition Rate (WRR), Word Error Rate (WER), which are calculated using the following formulas:

$$WER = \frac{S + I + D}{T} \tag{1}$$

$$WRR = 1 - WER \tag{2}$$

where S - the number of word replacement operations, I - the number of word insertion operations, D - the number of operations to delete words from the recognized phrase to get the original phrase, T - the number of words in the original phrase and is measured as a percentage.

In terms of recognition speed, the comparison was carried out using the Real Time Factor, an indicator of the ratio of recognition time to the duration of the recognized signal, also known as the Speed Factor (SF). This indicator can be calculated using the formula:

$$SF = \frac{Treg}{T} \tag{3}$$

where Treg - the time of signal recognition, T - its duration and is measured in fractions of real time.

All systems were trained using the WSJ1 (Wall Street Journal 1) speech corpus, which contains about 160 h of training data and 10 h of test data representing excerpts from the newspaper "Wall Street Journal". This speech corpus is usually taken for training purposes and includes recordings of male and female speakers in English.

After conducting the experiment and processing the results, the following table was obtained (Table 1).

Table 1. Results of comparison in accuracy and speed

System	WER	WRR	SF
HTK	19,5	80,5	1.4
Kaldi	6.6	93.4	0.6
Julius	23.2	76.8	1.4
RWTH ASR	15.4	84.6	3.8

At the same time, the accuracy and correctness of the study is confirmed by the fact that the results obtained are similar to those obtained when testing these systems on other speech corpora, such as Verbmobil 1, Quaero, EPPS [12,17,18].

To compare the structures, the system implementation language, algorithms used in recognition, input and output data formats and the internal structure of the software implementation of the system were selected as criteria, which is reflected in the tables (Tables 2, 3).

From the point of view of usability, the following indicators were considered - support for various software and hardware runtimes, support for many natural recognition languages, interface characteristics, availability of documentation, license restrictions. The results are presented in the following tables (Table 4).

During the experiment and subsequent analysis, the Kaldi system was selected. Which shows the best recognition accuracy of all the systems under

Table 2. System implementation languages and their structure

System	languages	structure
HTK	C	Modular, in the form of utilities
Kaldi	C++	Modular
Julius	C	Modular
RWTH ASR	C++	Modular

Table 3. Algorithm comparison results

System	Feature extraction	Acoustic modeling	Language modeling	Recognition
HTK	MFCC	HMM	N-gramm	The Viterbi algorithm
Kaldi	MFCC, PLP	HMM, GMM, SGMM, DNN	FST	Two-pass forward-reverse algorithm
Julius	MFCC	HMM	N-gramm, Rule-based	The Viterbi algorithm
RWTH ASR	MFCC, PLP	HMM, GMM	N-gramm, WFST	The Viterbi algorithm

consideration (WER = 6.6) and the second recognition rate (SF = 0.6). In terms of the algorithms and data structures provided for speech recognition, this system is a leader, as it provides the largest number of modern approaches used in the field of speech recognition, including the use of neural networks.

In addition, it allows you to use a variety of algorithms to reduce the size of the acoustic features of the signal, and, accordingly, increase the performance of the system. Kaldi is written in the C++ programming language, which has a

Table 4. The main indicators for SRS

System	Supported OS	Languages	Interface	Documentation	License
HTK	Linux, Mac OS, FreeBSD, Windows	English	Console, API	Comprehensive documentation	HTK
Kaldi	Linux, FreeBSD, Windows	English	Console	Detailed online documentation	Apache
Julius	Linux, Mac OS, FreeBSD, Windows	English, Japanese	Console, API	Comprehensive documentation	BSD like
RWTH ASR	Linux, Mac OS	English	Console	Not detailed documentation	RWTH ASR

positive effect on the speed of the system, and has a modular structure, which makes it possible to easily refactor the system, add new functionality, and correct existing errors. From the point of view of ease of use of Kaldi, it is necessary to note the availability of detailed documentation. The system is cross-platform, but provides only a console interface, which makes its integration into third-party applications difficult, but real. By default, this system supports only English (there are possibilities to connect other languages) and is distributed under a completely free Apache license, which allows it to be integrated into a commercial product without disclosing its code.

To build the SPP, you need to install Python v.3.8, the PyAudio = 0.2.11 and Vosk = 0.3.1.2 libraries, where Vosk is a stand-alone tool for speech recognition with open source code and a set of a large number of language models. When working with the Vosk model, the sampling frequency must be correct (a model with a large number of recognition languages supports a frequency of 8000, therefore, data from the microphone must be read at the same frequency).

It should be noted that the Vosk model functions correctly in the absence of noise, so it is necessary to either remove noise effects and extraneous sounds (there are a number of methods developed for this) [19], or use an alternative model that does not require preliminary cleaning from noise interference.

To significantly improve the speech potential of the "Smart Ecosystem", it is necessary to use the capabilities of AI-based technologies, given that the Kaldi application option involves connecting neural networks.

6 Option to Build a "Smart Ecosystem" Using Speech Input Technology for Control and Configuration

Taking into account the analysis of the options for building "Smart systems" and a number of research experiments, a block diagram of 4 software components (SC) is proposed (Fig. 3).

Thus, the proposed CPP includes:

1. SC "INFORMATION INPUT AND RECOGNITION" consisting of modules:
   - Data entry module - getting sounds into a recognition software application;
   - Data Preprocessing module - processing of the received audio signal, quality assessment and elimination of interference, removal of low-quality data, missing values, sound distortion of emissions);
   - Data Recognition module - (allocation of signal sections containing speech training commands using keywords - speech recognition in selected areas, followed by saving in the database.
2. SC "ORGANIZATION OF INFORMATION STORAGE (DB)" consisting of modules:
   - Equipment Reference - designed for storing a list of equipment with settings;
   - Command Reference - designed to store a list of configuration commands;

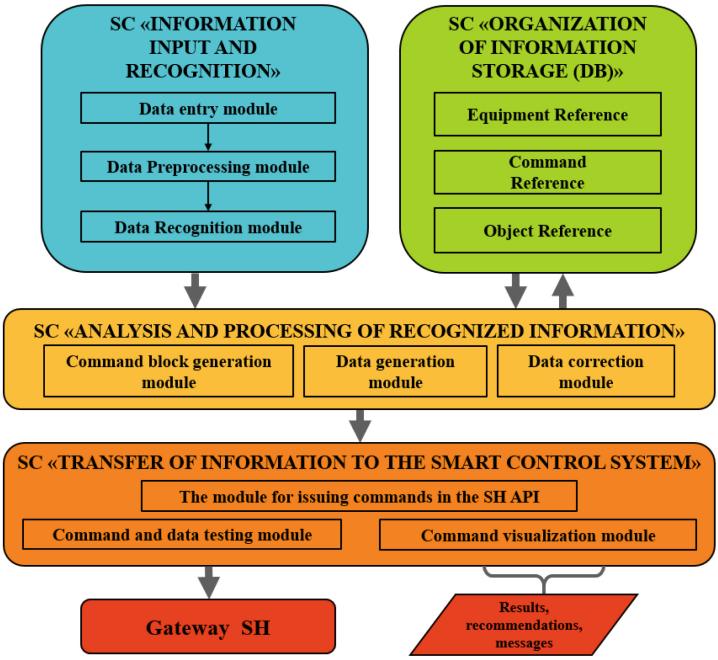

Fig. 3. Block diagram of organizing speech input into the "Smart Ecosystem"

- Object Reference - designed for storing a list of objects with settings.

3. SC «ANALYSIS AND PROCESSING OF RECOGNIZED INFORMATION» consisting of modules:
   - Command block generation module - processing and analysis of the information received, compilation of team buildings;
   - Data generation module - checking the correctness of the names of objects, commands, and the installation position of equipment;
   - Data correction module - correction of distorted information, correction if possible, request for re-entry.
4. SC «TRANSFER OF INFORMATION TO THE SMART CONTROL SYSTEM» consisting of modules:
   - The command and data testing module - issuing commands to verify compliance;
   - The module for issuing commands in the Smart Environment API - issuing commands to configure the system;
   - Command Visualization Module - issuing commands for system configuration.

The proposed SSP has expansion points for connecting software modules with AI-based capabilities, but at the current stage of development, it is tentatively planned to implement only voice input with AI.

Among the many capabilities of artificial intelligence technologies in the field of SSP, AI voice assistant as software that applies artificial intelligence and natural language processing to interact with users through speech is the first to be highlighted. With the help of available libraries and tools, it is easy to set up an environment and create a voice assistant that will respond to user requests and perform various tasks.

In this case, AI voice assistant can be used:

1. To improve recognition accuracy when entering configuration and training commands into the "Smart Ecosystem" (reducing noise levels, identifying dialects and normalizing the sound level, checking for correctness of input and compliance with the configuration of the created "Smart System", etc.).
2. To transform the input stream to a unified standard and generate a unified control and training effect.
3. To expand the functionality of the training system for adjustments and prepare additional capabilities for the "Smart Ecosystem".
4. To develop recommendations for managing Smart Ecosystem devices.

This list can be continued, but it is necessary to understand to what extent we want to automate processes in the "Smart Ecosystem" and what resources we have at our disposal. Thus, the creation and use of an AI voice assistant is a promising project that can provide users with many new opportunities.

7 Conclusion

The research conducted by the authors of the article aimed to show the possibility for the Customer (User) to create for themselves a single "Smart Ecosystem".

The main problematic issues of constructing and configuring automation systems based on the "Smart Environment" technology with the possibility of combining heterogeneous equipment from different manufacturers in a single information space are considered.

This work proposes a sequence of steps and a set of technologies that make it possible to implement an SSP with the possibility of speech input of control commands for setting up and further training the "Smart Environment" installed on the object. The use of AI technology for training "Smart Systems" allows the customer to significantly expand the capabilities of their personal "Smart Ecosystem", as well as independently configure and subsequently train this system without serious professional training.

References

1. Bogdanov, A., Shchegoleva, N., Dik, G., Khvatov, A., Dik, A.: Smart Habitat: Features of Building it Infrastructure, Main Problems of Building Data Networks Using 5G (6G) Technologies
2. Barashko, E.N., Vasiliev, A.S., Zuban, S.V.: Voice assistants. New Impulses Dev. Issues Sci. Res. **1**, 47–53 (2020)

3. Dik, G., et al.: New Security Challenges of Internet of Things. In: International Conference on Computational Science and Its Applications, pp. 302–316. Springer, Cham (2023)
4. Hammi, B., et al.: Survey on smart homes: vulnerabilities, risks, and countermeasures. Comput. Secur. **117**, 102677 (2022)
5. Baum, L.E.: Statistical inference for probabilistic functions of finite state Markov chains. Baum L. E., Petrie T., Ann. Math. Stat. **37**, 1554–1563 (1960)
6. Baum, L.E., Petrie, T., Soules, G., Weiss, N.: A maximisation technique occurring in the statistical analysis of probabilistic functions of Markov Chains. Ann. Math. Stat. **41**(1), 164–171 (1970)
7. Oppen-hejm, A.V., Schafer, R.W.: From frequency to quefrancy: a history of the cepstrum. IEEE Signal Process. Mag. **21**, 95–106 (2004)
8. Rabiner, L.R., Juang, B.N.: Fundamentals of Speach Recognition, New Jersey (1993)
9. Huang, L., et al.: A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. arXiv preprint arXiv:2311.05232 (2023)
10. Obaid, A.J.: Assessment of smart home assistants as an IoT. Int. J. Comput. Inf. Manuf. (IJCIM) **1**(1) (2021)
11. Ravanelli, M., Parcollet, T., Bengio, Y.: The pytorch-kaldi speech recognition toolkit. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6465–6469. IEEE (2019)
12. Gaida, C., et al.: Comparing open-source speech recognition toolkits [Ectronic resource]. Technical Report of the Project OASIS. http://suendermann.com/su/pdf/oasis2014.pdf (date of application: 03.05.2025)
13. El Moubtahij, H., Halli, A., Satori, K.: Using features of local densities, statistics and HMM toolkit (HTK) for offline Arabic handwriting text recognition. J. Electr. Syst. Inf. Technol. **3**(3), 99–110 (2016)
14. Ogata, K., Nakashima, K.: Analysis of articulatory timing based on a superposition model for VCV sequences. In: Proceedings of IEEE International Conference on Systems, Man and Cybernetics, pp. 3720–3725, January 2014
15. Belenko, M.V.: Sravnitelnyy analiz sistem raspoznavaniya rechi s otkrytym kodom [analysis and comparison of the open source speech recognition systems] Sbornik trudov V Vserossiyskogo kongressa molodyih uchenyih [Collection of Proceedings of the V All-Russian Congress of Young Scientists]. V. 2. - SPb.: ITMO University, 2016, pp. 45–49. [in Russian]
16. Karpov, A.A., Akarun, L., Ronzhin , A.L.: Mnogomodalnyie assistivnyie sistemyi dlya intellektualnogo zhilogo prostranstva [Multi-modal assistive systems for intelligent living space]. Trudyi SPIIRAN [Proceedings of SPIIRAS], 19, pp. 48–64 (2011). [in Russian]
17. Jha, M., et al.: Improved unsupervised speech recognition system using MLLR speaker adaptation and confidence measurement. V Jornadas en Tecnologıas del Habla (VJTH'2008), pp. 255–258 (2008)
18. Sundermeyer, M., et al.: The rwth 2010 quaero asr evaluation system for English, French, and German. In: Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2212–2215 (2011)
19. Zhao, W.X., et al.: A survey of large language models. arXiv preprint arXiv:2303.18223 (2023)