Creation of a platform enabling the use of the latest information technology in embedded devices

Yoshiaki Komoriya, Sota Nakabayashi, Isao Yamada

Key words

savic-net G5, building air conditioning system, comfort, energy efficiency, control application, artificial intelligence, Python

In recent years, in order to realize further comfort and energy savings from building air conditioning systems, the need for artificial intelligence and other advanced information technology has become apparent. However, a new control application for the embedded devices incorporated into a building system usually must be created using the already-implemented control application language, making it difficult to utilize the latest information technology. To solve this problem we created a platform that allows control applications written with general-purpose programming languages and libraries to be added to embedded devices. Existing embedded devices can now take advantage of the latest information technology without the need to add a PC to the system.

1. Introduction

Buildings are equipped with many different facilities, including central plant equipment and air conditioning equipment. In the air conditioning system, embedded devices called controllers link the devices in various types of equipment and control them for energy efficiency and comfortable air conditioning throughout the building. A controller is more reliable, consumes less power, and is more efficient in terms of space than a common PC.

Azbil provides a wide range of standard control functions in its building air conditioning systems, and can also develop control applications customized to the customer's requirements and add them to embedded devices.

In general, control applications added to embedded devices are developed in a language dedicated for that purpose (as defined, for example, by the international IEC 61131-3 standard) rather than a general-purpose programming language like Java, JavaScript, Python, C#, Go, or R [1]. A programming language dedicated for control was also used to develop control applications in Azbil's previous building air conditioning systems like savic-net FX™. Such a language is designed for embedded devices, which have more stringent restrictions on CPU performance and memory space than is the case for PCs.

Although AI and other relatively new types of information technology are being used in a variety of fields, languages dedicated to control have lower functionality and less freedom than general-purpose programming languages and are not suitable for advanced scientific computing or implementation of complicated logic.

In addition, because many people use general-purpose programming languages, there is constant functional improvement and development of libraries in order to use the latest information technology, but again this is not the case for languages dedicated to control.

In other words, it is difficult to use the latest IT in languages dedicated to control, and separate measures are required to do so. For example, it may be necessary to add PCs to the system in order to execute control applications developed in a general-purpose programming language.

To solve this problem, we have created a platform that enables us to add control applications developed in a general-purpose programming language to embedded devices and to add new control applications that use the latest version of a language and the latest libraries without inhibiting the functions of existing control applications.

With this platform, control applications that use the latest information technology can be added to existing embedded devices without incurring the cost of adding new PCs to the building air conditioning system.

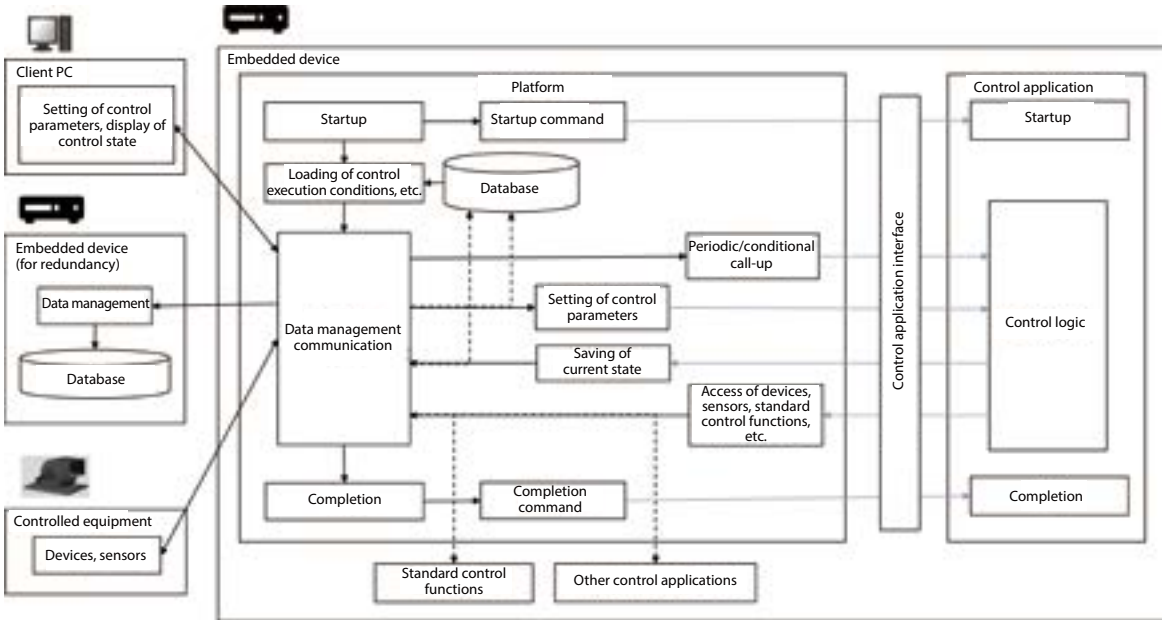

Fig. 1. Overview of the platform

2. Development of control applications in general-purpose programming languages

2.1 Overview of the platform

An overview of this platform is shown in figure 1. This platform and the control applications operate as processes independent of one another. When a device is turned ON, the platform starts up and orders each control application to start up. Then, the platform loads control conditions and other data and calls up the control logic implemented in each control application in the specified cycle or when specified conditions are met. When the device is turned OFF, the platform orders each control application to shut down.

The platform tracks the control parameters for each control application set on the screen on the client PC and the current state of each control application in the database. If the embedded device is configured for redundancy, the platform synchronizes data with the database in the redundant embedded device in real time. This allows each control application in the redundant embedded device to take over the control parameters and the current state in order to continue control even if there is an unexpected failure that prevents the issuing of a shutdown command to each application.

The platform also provides a function for communicating with the devices and sensors in the controlled equipment so that the control application can access the states of the devices and the measurement values of the sensors. As a result, the control application can read and write the states and measurement values of the devices and sensors regardless of their location or the communication protocol used for access (such as BACnet or Modbus). This platform also provides the functions for control applications to access the control parameters and current state of the standard control functions in embedded devices and other existing control applications.

The platform provides these functions for each control application via the control application interface. This interface consists of a protocol that is independent of the programming language (the WebSocket protocol) and a message format (JSON). This configuration permits the development of control applications not only in some specific programming language but also in many different general-purpose languages.

2.2 Virtualization of the control application execution environment

The functionality of many general-purpose programming languages is constantly being improved, and new versions of languages and libraries appear one after another. Version compatibility is not always maintained when new versions appear. In other words, if a language or library embedded in the device OS is upgraded when a control application developed in a new version of the language or library is being added to an embedded device, the existing control applications developed in an old version may stop operating.

To avoid this problem, this platform implements virtualization technology for embedded devices so that adding an application developed in a new version of a language or library does not affect existing control applications. Specifically, the virtualization technology isolates the execution environment of the control application. The control application itself and the language and library that are used are added into the execution environment, which is isolated from the device OS rather than embedded in it (fig. 2). With this mechanism it is possible to add newly developed control applications using the latest version of a language or library.

2.3 Reduction in memory usage

It is more difficult to enhance the CPU performance and expand the memory space in an embedded device than in a server, PC, or other system. In addition, because a general-purpose programming language has higher functionality and greater freedom than a language dedicated to control, a control application created in a general-purpose programming language tends to use more memory than one created in a dedicated control language.

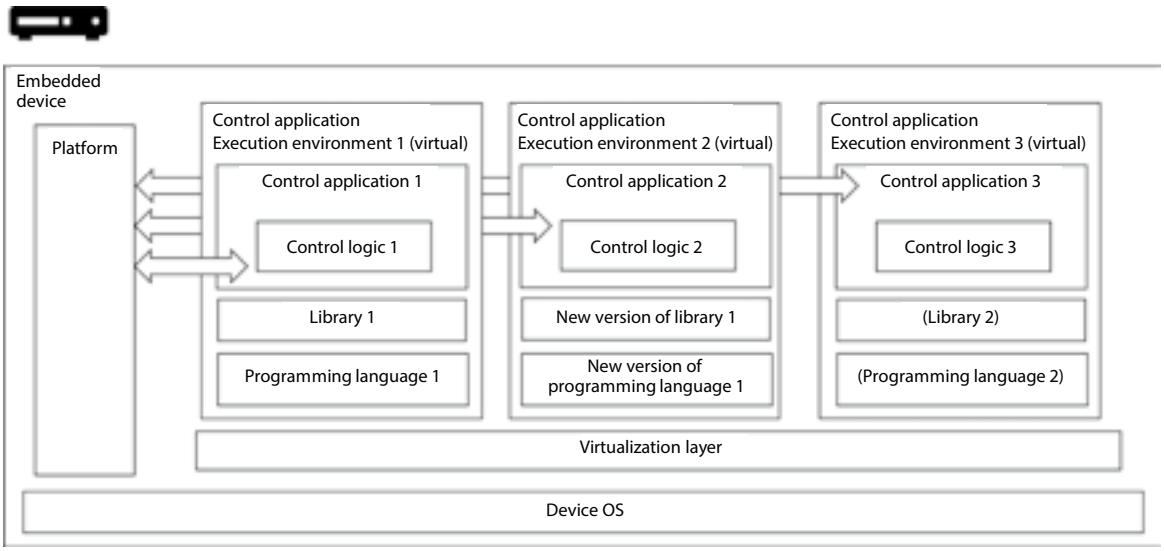

Fig. 2. Virtualization of the control application execution environment

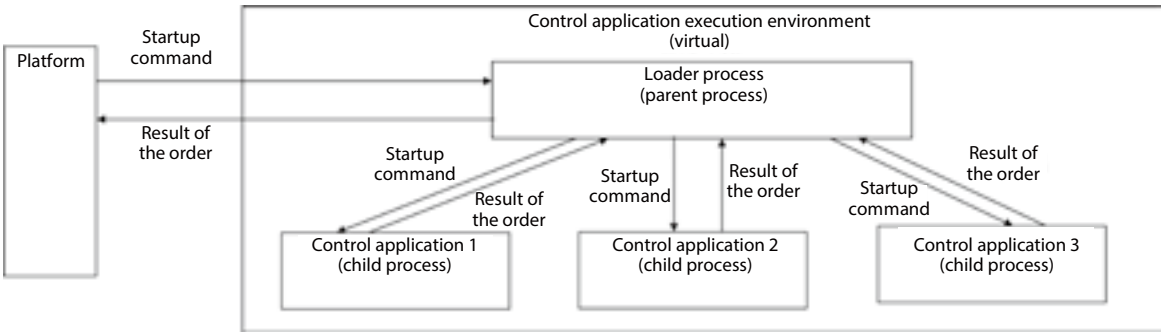

Fig. 3. Control application startup mechanism

We have adopted container virtualization technology [2], which requires less CPU performance and memory space than host or hypervisor virtualization technology, in order to provide virtualization in embedded devices, since these devices have stringent restrictions on CPU performance and memory space.

With container virtualization technology, it is said that each application should be executed in its own application execution environment to ensure maintainability and manageability. However, strictly adopting this method would require a large amount of memory, since the same amount of memory specific to each control application execution environment would have to be reserved for each executed control application. In other words, only a small number of control applications can be operated in an embedded device with stringent memory constraints. Therefore, on this platform, control applications developed with the same language and library versions are executed in the same control application execution environment.

To further reduce memory usage, we developed a mechanism for starting multiple control applications that are executed in the same execution environment as child processes from the loader process that imported the necessary library (fig. 3).

Usually, a library is loaded into an independent memory space specific to each control application. However, the mechanism we developed allows each control application to share the memory space where the loader put a library. We found, for example, that the mechanism reduces memory usage by about 50% when 30 control applications developed in Python are running (fig. 4).

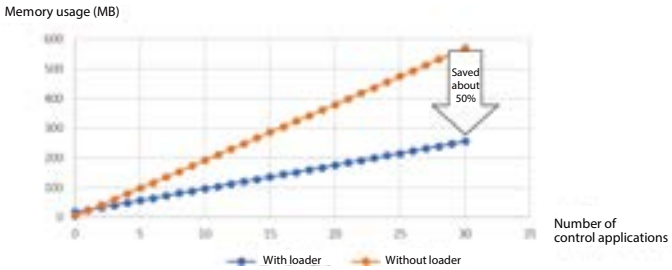

Fig. 4. Comparison of memory usage

3. Commercialization of the technology

3.1 Adoption of Python as the general-purpose programming language

In December 2020, Azbil launched the savic-net™ G5 Supervisory Controller [3]. This is an embedded device that can add control applications developed in Python to Azbil's latest building air conditioning system, savic-net™ G5, using the platform (fig. 5).

Fig. 5. savic-net™ G5 Supervisory Controller

Python is a general-purpose programming language widely used in many different fields, including fields applying artificial intelligence applications, because it is easy to learn, enables high coding productivity, and supports the implementation of advanced scientific computing and complicated logic. Now that control applications can be developed in Python, the Python library can be used to efficiently perform complicated scientific computations and to implement machine learning in control applications. In addition, commercially available Python development environments with rich code-editing functions and debugging functions can now be used to develop control applications.

3.2 Control application development support tool

In addition, not only can programmers develop control applications in Python, but they can also use a development support tool to further facilitate development (fig. 6).

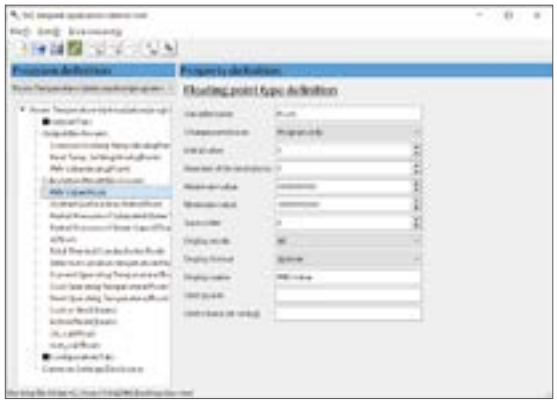

Fig. 6. Control application development support tool

The control application developer can enter information about the application into the tool, such as the application's name and control conditions, as well as the data names and types of the variables, etc., so that the control parameters and current values can be stored. During control, for example, the tool automatically outputs the functions to be called from the platform and the data structures and the definitions of variables stored in the database table to the Python code source file or other file based on the entered information. The tool also automatically generates a screen so that the control application user can check the settings and current states of control parameters based on the entered information [4].

With this control application development support tool, application developers no longer have to manually define data structures or create screens and other components. All they need to do to develop a control application is to add control logic to the functions in the source file output from the tool.

4. Examples of control application development

4.1 Optimal room temperature setting application

This section describes an optimal room temperature setting application as an example of a control application created with this platform (fig. 7).

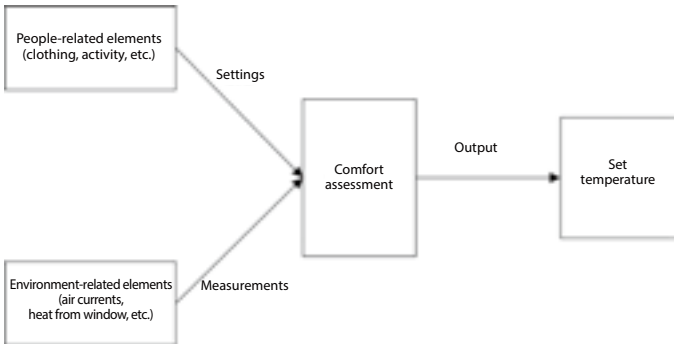

Fig. 7. Optimal room temperature setting application

In contrast to the usual air conditioning control where people decide what temperature to set, this application dynamically and automatically calculates the temperature at which people in the room will feel comfortable based on people-related factors like the amount of clothing they are wearing and their activity level, as well as environment-related factors such as the air currents in the room and the heat from the windows. In this way, the application avoids setting a room temperature that is too low or too high while automatically providing both comfort and energy efficiency.

Measurements usable as environment-related factors differ depending on how the sensors are installed in the room and other circumstances. In addition, there can be an infinite number of variations for comfort assessment and determination of the set temperature, including methods using advanced scientific computing. Even for this kind of control, advanced requirements specific to each customer can be addressed flexibly by developing a control application in Python on this platform.

4.2 Use in an artificial intelligence control demonstration experiment

We also prototyped an application that learns from past data obtained from controlled equipment and used the obtained learning model for control in an attempt to apply artificial intelligence (fig. 8).

With this platform and the development support tool, control application developers can focus on figuring out and implementing control logic and can leverage Python libraries to develop control applications efficiently and to verify the effects of the control logic.

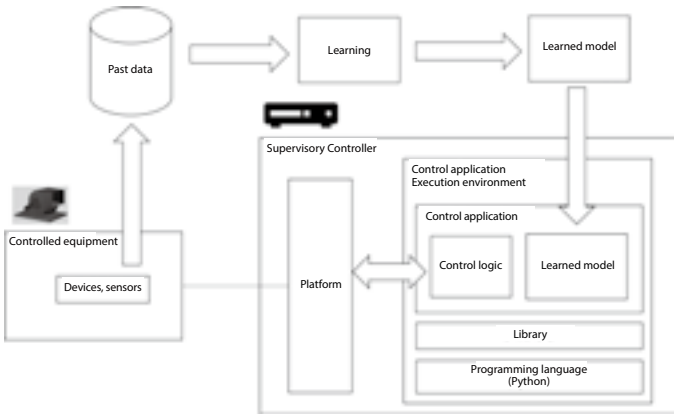

Fig. 8. Application example in the artificial intelligence field

5. Conclusion

The platform described in this article allows developers to add control applications written in the latest general-purpose programming language and with the latest libraries to embedded devices. In this way, the latest IT can be used in existing embedded devices without having to add PCs or other equipment to the system.

In addition, the use of Python and the control application development support tool facilitate the development of control applications.

In the future, we intend to establish a system for outside parties such as dealers to develop control applications. We would also like to promote projects such as artificial intelligence control demonstration experiments using this technology. We hope that this technology will accelerate the use of artificial intelligence and other advanced IT in building air conditioning systems.

References

[1] The Japan Electrical Manufacturers' Association. "Guidelines for improving the development efficiency of PLC applications" (in Japanese), available at https://www.jema-net.or.jp/jema/data/plc_ver1.pdf.

[2] Kawaguchi, Naoya. *Introduction to container virtualization* (in Japanese). CUTT System, 2020.

[3] Fukaura, Tsutomu. "The savic-net™ G5 Supervisory Controller for advanced energy management and comfortable indoor spaces" (in Japanese). *azbil Technical Review* (April 2019), 17–21.

[4] Nishira, Daiki, Hiroshi Koga, and Hiroyuki Yamamoto. "A new building management system offering an excellent user experience" (in Japanese). *azbil Technical Review* (April 2016), 27–33.

Author affiliation

Yoshiaki Komoriya — Development Department 1, Development Headquarters, Building Systems Company, Azbil Corporation

Sota Nakabayashi — BA Engineering Department, Building Systems Company, Azbil Corporation

Isao Yamada — AI Solution Department, Azbil Corporation