IoT Data Collection from Control Devices

KOBAYASHI Yoshiaki

In recent years, it has become important to make a full use of sensing data from control devices like sensors via new IT technology such as AI. To accelerate this, it is important to collect and transform ill-formed sensing data from control devices to well-formed data.

In this paper, we have tried to transform raw data form various control devices to IoT data systematically by using PLC in FA field as a data hub.

As an implementation, we have developed "Device Information Library" which contains specifications of devices and "Formatting Tool" software which generates PLC programs to collect and convert data based on Device Information Library. The generated programs transferred into a PLC execute integrally data collection, transformation and upload to IT layers.

1. Introduction

In factory automation (hereinafter "FA"), many different control device interfaces have been established to suit the intended uses or technological levels of the times. Typical interfaces include traditional analog I/O interfaces or recent communication interfaces such as IO-Link interfaces. From among these, interfaces with the optimal function and performance for devices or purposes on FA sites have been selected. It is not rare that control devices are kept in service for a long time; even for new facilities, old but field-proven control devices are often purposely selected with priority given to reliability. Consequently, it is usual at FA sites that different connection interfaces are in mixed use rather than dramatically converge into one type through survival of the fittest.

To ensure the functioning of such a wide range of different types of control devices, it is common to write a control program, with control devices (hereinafter simply referred to as "devices") connected to a general-purpose device such as a programmable logic controller (hereinafter "PLC") or an industrial PC (hereinafter "IPC"); their programming requires different knowledge for each interface.

In addition, among devices with the same type of interface, each device may differ from the others in characteristics such as unit or number of significant digits. Moreover, programming is highly flexible and hence ends up with dependency on individual skill. Due to such diversity and dependency on individual skill, data formats generated from FA sites tend to be inconsistent, thus posing an obstacle to secondary use of control data for analysis and other purposes.

In traditional system development, coordination among devices around a controller, such as a PLC or an IPC, was generally limited to within the same system. Hence, inconsistencies in data format were not very often regarded as problematic. Because relevant devices used to be limited to those directly involved in control, the burden on engineers was not so heavy as today. The dependency of performance and quality on individual skill was considered to give an edge to integral-type development rather than otherwise[1].

Amid the accelerating efforts towards the fourth industrial revolution[2], however, it is increasingly necessary to collect an unprecedented amount of sensing data, including those not directly related to control, for use in transversal analysis, whether inside or outside the system. Typical examples of such sensing data include those on overall equipment effectiveness, energy-saving, and traceability. Omron has also started up its own IoT service platform "i-BELT" to accelerate the above-mentioned efforts[3].

For such purposes, it is required to collect data not in various "raw" formats as seen on-site but in a standardized format containing meta-information, such as units and accuracies, which facilitates analyses and AI processes on the IT layer[4]. It is difficult to narrow down the scope of data collection in advance because the usability of individual data usually cannot be known before analysis. This leads to an increase in the programming costs for data collection from many sensors, posing a major obstacle to starting information utilization activities.

This paper presents an approach to automatically generating data collection-conversion programs from the archived spec information of devices, to use a PLC as a data hub for the collective transmission of data collected from devices in order to convert raw data from various control devices, including analog sensors, systematically into IoT data.

Contact : *KOBAYASHI Yoshiaki* yoshiaki.kobayashi@omron.com

2. Challenges

2.1 Differences between OT- and IT-based data models

To take full advantage of AI and other IT technologies for effective use of data collected from control devices, it is common practice to transfer such data to IT-based storage and/or interfaces such as relational databases, data lakes, and message brokers. For transmission and reception, it is necessary that data models sent from a PLC or an IPC be the same as those on the IT/ reception side. PLC and IPC programs are written by engineers skilled in the so-called operational technology (OT) whereas programs used on the reception side are written by IT-savvy engineers.

Thus, IoT data exchange involves cross-field communication between OT and IT engineers. The differences in terminology and knowledge between them lead to delayed data exchanges and to inconsistencies in exchanged data, which can easily result in problems such as use of the wrong data.

Additionally, pieces of hardware on FA sites are designed by FA engineers using CAD and PLC tools. Hence, their design information includes device configuration information such as device connection information (hereinafter "device configuration information"). To ensure proper transmission of OT-side information to the IT side, IT-side data models should be created using design information as primary information; technical and methodological differences are, however, disrupting the flow of information.

2.2 Inconsistencies in data format

When relying on a program for data collection, the format of collected data and the meta-information added thereto may vary depending on the program implementer. Typical differences include those in data item names, value types and accuracies, and unit notations.

It is possible to achieve standardization based on standards or protocols in an industry or organization. Standards and protocols, however, differ from one industry/organization to another. Whether or not to comply with them is left to the discretion of implementers, and it is far from easy to ensure full enforcement of compliance with them. In the IoT field, cross-industrial or cross-organizational data collection is widely practiced[5], adding more difficulty to format standardization.

2.3 Programming costs

A PLC or an IPC can rely on programs for different types of processes to meet a wide range of user needs. This, however, means that PLCs/IPCs need a program even for a process as simple as data collection. For the development of such a program, it will be necessary to read the spec information of the interface or sensor connected to the PLC/IPC from their data sheets and to turn such information into built-in parameters.

Such data collection-conversion programs are quite similar to one another, and therefore can be made routine to some extent by skilled engineers. Even so, when such programs are intended for a large number of devices, the incurred cost will not be negligible. In addition, programs developed by unskilled engineers will pose a higher risk of performance and quality problems.

3. Technical details

3.1 Overview

We have developed a "Device Information Library" as the registry of the spec information of control devices and a "Formatting Tool," software capable of automatically generating PLC programs for data collection-conversion from information obtained by referring to the Device Information Library. Typical control devices to be registered in the "Device Information Library" include sensors as well as I/O units and communication units for connecting them to a PLC.

The Structured Text (ST) language defined in the international standard IEC 61131−3 was adopted as the programming language for programs to be generated, considering that it is text-based, hence highly compatible with automatic generation technology, and well-suited for the logic operations and numeric calculations necessary for conversion processes[6].

A program generated by the Formatting Tool collects data from devices connected to a PLC and stores the collected data in a structure named an "industrial quantity variable." Data obtained from devices are stored in a PLC memory area called a "device variable." Their values are usually substitution values such as AD conversion values rather than industrial quantities *per se*. In such cases, industrial quantities are first converted into industrial values in the generated program and then stored in the industrial quantity variable. The industrial quantity variable can serve as a network variable, which provides readouts from outside the PLC and uses the PLC's database connection function for transmission to external databases.

The system configuration is shown in Fig. 1, followed by the description of the flow from program generation to data collection.

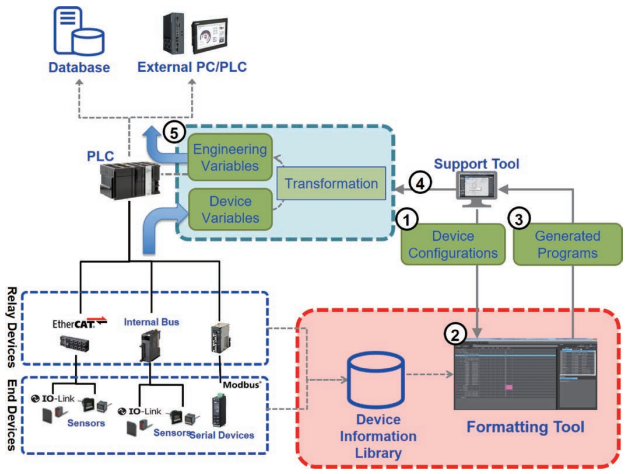

Fig. 1 System configuration

(1) The device configuration information covering up to the relay devices (listed in 3.2) set up using the PLC Support Tool (hereinafter "Support Tool") is loaded into the Formatting Tool;

(2) In accordance with the user-selected operation, control device-related information including industrial quantity information, conversion information, and unit information is added from the Device Information Library to (1) to complete a device map (described in 3.2);

(3) A program is automatically generated from the completed device map;

(4) The automatically generated program is transferred via the Support Tool to the PLC; and

(5) The transferred program is run on the PLC to collect data from devices, make the data accessible via the network, and transmit them to a database(s).

3.2 Device map creation

The Support Tool manages the information on I/O units and communication units (hereinafter "relay devices") connected to the PLC as device configuration information. The Formatting Tool loads the device configuration information from the PLC. Then, the Formatting Tool writes the information to terminal devices, such as sensors, downstream to the relay devices, and sets up the terminal devices to create a device map. Fig. 2 shows a conceptual schematic of such a device map.

The device map is defined by a tree structure, with a PLC being the starting point and individual devices being nodes; these individual devices are structured to have more than one input and output. This structure is not dependent on any connection interface, and connection interface differences are registered in the Device Information Library as attributes of individual inputs and outputs. The Formatting Tool builds a device map in such a manner that the interface to the input of each relay device corresponds with that of the output of each terminal device.

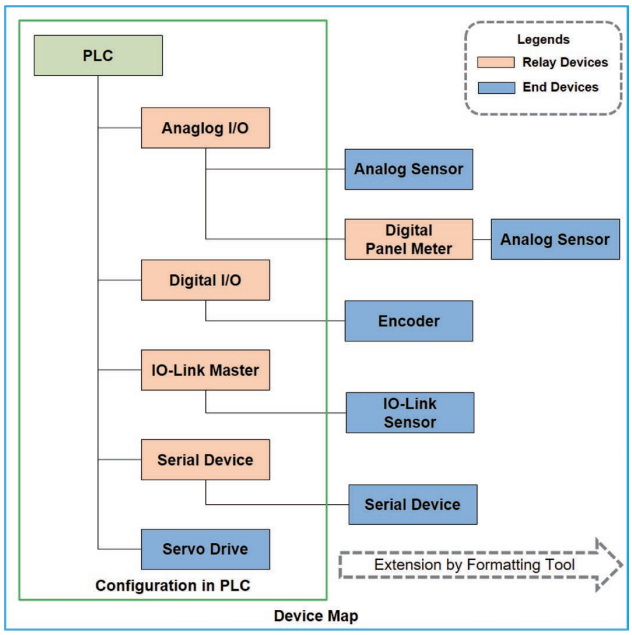

Fig. 2 Conceptual schematic of a device map

Let us explain a typical case of device selection by referring to the simplified schematic of the Device Information Library in Fig. 3 and the screenshots of the Formatting Tool in Figs. 4 and 5. From the device configuration information possessed by the Support Tool, the library is searched using the formats of the relay devices as the search conditions ((1)) to show the user the analog and other interfaces of the inputs of the relay devices ((2)). When the user selects from these interfaces, a list of devices having a spec-matched interface and an industrial quantity ((3)) as the output will be presented to the user as the list of candidate options. When the user selects one of the candidate options ((4)), the parameters of that device will be obtained to identify the industrial quantity to be measured ((5)).

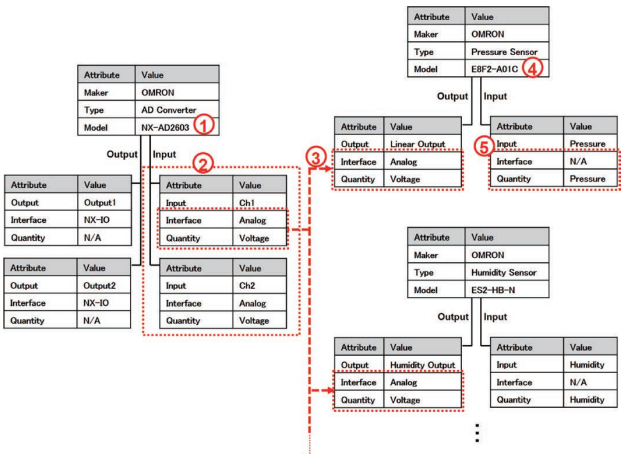

Fig. 3 Device selection from the Device Information Library

Fig. 4 Presentation of devices available for selection

Fig. 5 Identification of the relevant industrial quantity

3.3 Separation into a common template program and device-specific parameters

To generate data collection-conversion programs that suit individual devices, for each connection interface, a common template program and device-specific parameters are separately registered in the Device Information Library.

The following explains this using an analog sensor as an example. This analog sensor measures and converts an industrial quantity into a voltage or current output. The AD conversion unit converts the analog sensor output into a digital value. The PLC program converts this digital value into a voltage or current value according to the specifications of the AD conversion unit, and then performs another conversion according to the specifications of the analog sensor to obtain the industrial quantity. In the example shown in Fig. 6, a measured value of 50 kPa is stored as a conversion value of 1200 via a voltage value of 3 V in the device variable of the AD conversion unit.

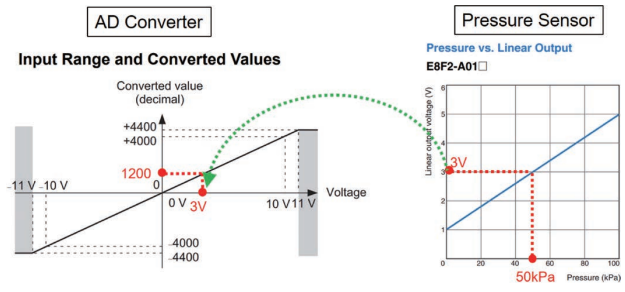

Fig. 6 Loading of a measured value

To convert the value of the device variable into pressure (unit: kPa), which is an industrial quantity, the linear conversion formulae (1) and (2) shown in Fig. 7 are used. The meanings of the variables are as shown in Table 1. Pressure is the industrial quantity variable.

```
Voltage :=
  (Analog_Input_Value + 4000) / 8000 * 20 — 10; ...(1)
Pressure :=
  (Voltage - 1) / 4 * 100; ...(2)
```

Fig. 7 Conversion into an industrial quantity

Table 1 Variables for use in conversion formulae

Variable name	Content	Unit	Typical value
Analog_Input_Value	Device variable	None	1200
Voltage	Voltage value	V	3
Pressure	Pressure value	kPa	50

While these linear conversion formulae are common for analog sensors and AD conversion units, the parameters for determining the tilt and offset differ from one device to another. Then, to generate conversion formulae such as (1) and (2), the common template program shown in Fig. 8 is registered in the Device Information Library. The template program has built-in parameters, each contained in ${ }. Table 2 shows a list of parameters for linear conversion formulae.

```
${output} :=
    (${input} - ${outMin}) / (${outMax} - ${outMin}) *
    (${inMax} - ${inMin}) + ${inMin};
```

Fig. 8 Template program for linear conversion formulae

Table 2 List of parameters for linear conversion formulae

Parameter name	Content	Meaning
input	Variable name	Pre-conversion value
output	Variable name	Post-conversion value
inMax	Numerical value	Input upper limit
inMin	Numerical value	Input lower limit
outMax	Numerical value	Output upper limit
outMin	Numerical value	Output lower limit

In the Device Information Library, the specific values of parameters based on the specifications of each device are registered. The Formatting Tool substitutes the parameters in the template program with these specific values to automatically generate conversion formulae. In the case of Fig. 8, the parameters for the AD conversion unit are substituted with the values in Table 3 to generate Formula (1) in Fig. 7; and the parameters for the pressure sensor are substituted with the values in Table 4 to generate Formula (2) in Fig. 7.

Table 3 Typical parameters for an AD conversion unit

Parameter name	Value
input	Analog_Input_Value
output	Voltage
inMax	10
inMin	−10
outMax	4000
outMin	−4000

Table 4 Typical parameters for a pressure sensor

Parameter name	Value
input	Voltage
output	Pressure
inMax	0
inMin	100
outMax	1
outMin	5

The example shown above is a simple case of parameter substitution. For program generation from the template program, however, a branching or an iterative process can be used in line with the values of parameters.

In this way, a cast process based on the presence/absence of signs, a process of decoding bit-string outputs from an encoder to extract measured values, and so forth can be generated from the common template program. Moreover, not only variable value conversion process programs but also communication programs can be generated. For example, let us assume a Modbus RTU: a template is created for command transmission and reception processes; then thereinto embedded are device-specific parameters registered in the Device Information Library, such as function codes and register addresses, to enable automatic generation of a serial communication program.

Using as examples the five connection interfaces in Table 5, common processes were turned into template programs and broken down into parameters to demonstrate that programs can be generated to support individual devices.

Table 5 Connection interfaces and common processes

Interface	Common process	Main parameters
Analog	Linear conversion	I/O upper and lower limits
Encoder	Bit connecting	Number of bits
	Decoding	Encoding method (BCD, Gray code)
IO-Link	Bit operation	Bit offset Bit length
	Cast	Presence/absence of signs
	Linear conversion	Tilt and offset
EtherCAT	Linear conversion	Tilt and offset
Modbus RTU	Command transmission and reception	Function code Register address Number of words
	Endian conversion	Endian type
	Cast	Presence/absence of signs
	Linear conversion	Tilt and offset

3.4 Parameter switching based on selected settings

To support devices whose built-in parameters for conversion formulae are variable depending on the settings, the Device Information Library is designed to allow changing parameters for each setting. For example, in the case of an AD conversion unit capable of switching between input specifications, a group of parameters such as the one shown in Table 6 is registered in the Device Information Library.

Table 6 Typical parameters variable depending on the setting

Setting name	Industrial qty	inMax	inMin	outMax	outMin
−10–10V	Voltage	10	−10	4000	−4000
0–5V		5	0	8000	0
1–5V		5	1	8000	0
0–10V		10	0	8000	0
4–20mA	Current	20	4	8000	0

By selecting a setting name via the Formatting Tool as shown in Fig. 9, an appropriate parameter is applied accordingly.

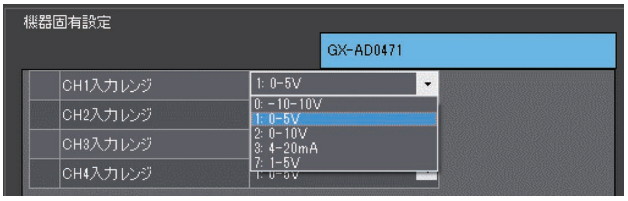

Fig. 9 Parameter determination based on selected settings

3.5 Data format standardization and transmission to external destinations

The structure in the PLC shown in Table 7 is automatically generated as the storage for an industrial quantity variable.

Table 7 Definition of structure of collected data

Item	Data type	Content
TimeStamp	DATE_AND_TIME	Collection time
FMT_<identifier>	LREAL	Industrial value
FMT_<identifier>_unit	STRING	Unit
FMT_<identifier>_ physicalQuantity	STRING	Industrial quantity name

The unit and industrial quantity, which are both the meta-information of collected data, are selected from a list in the Device Information Library and stored as character string types. This list is created in accordance with the notation rules specified in the SI Brochure and in the Japanese Measurement Act and Order for Measurement Units to prevent notational confusion due to personal preferences.

The internal clock of the PLC is used as the collection time, which is also meta-information, so that the collection times of all data will be recorded based on the same criterion.

If the PLC has a database connection function, the

Formatting Tool will generate a program that stores the value of the industrial quantity variable from the PLC to the database table in the host system. Additionally, the Formatting Tool also generates an SQL statement that defines the database table for the above storage. When this SQL statement is executed to the database, the data structures of the PLC and the database will be built synchronously with each other (See Fig. 10).

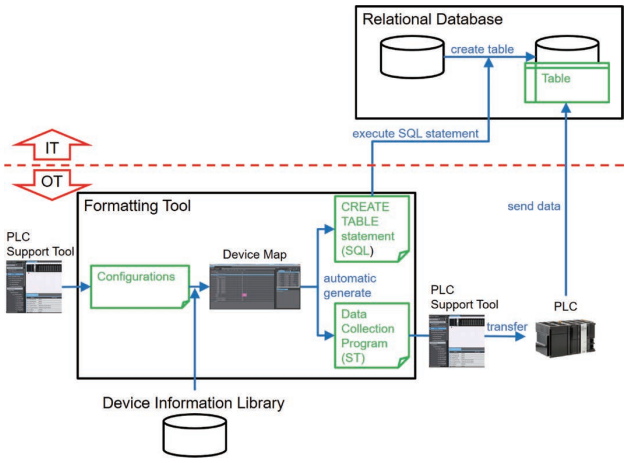

Fig. 10 Data flow in database transmission

4. Effectiveness

The Device Information Library can be said to be an archive of control device specifications, which are a type of OT domain knowledge. The device configuration information possessed by the Support Tool constitutes a part of system design information. Using this as the primary information, the Formatting Tool automatically generates standardized structures expressing information converted into industrial quantities, and also generates collection, conversion, and transmission programs therefor. Moreover, the Formatting Tool also automatically generates database table definitions as data models for IT layers.

In other words, the knowledge and skills of two different fields, namely, OT and IT fields have been combined into the above tools, whereby the same engineer can perform the complete process of transmission to a database. These tools are expected to help prevent delayed and/or inconsistent communications as mentioned in 2.1, while also reducing the disruption of information flow between the OT and IT sides.

The Formatting Tool automatically generates definitions of structures under certain rules in accordance with the content of the Device Information Library, and therefore provides identical results for identical device configurations. This enables standardization of data formats without depending on individual operators' skills, as described in 2.2.

Among data standardization technologies other than the one presented herein are a method that uses dedicated hardware, called a gateway device and usable as a data hub, or methods that perform conversion processes on IT systems including Clouds. The advantages of the approach presented herein over these alternatives include the following: it allows easy handling of data synchronized with production takt times or control periods because the PLC itself carries out data collection; and it allows use of system design information.

The Formatting Tool allows the user to identify the information necessary for program generation by performing the simple operations of selecting devices and settings from the Device Information Library when creating a device map.

Therefore, as explained in 2.3, unlike conventional methods the approach presented herein does not require spec information loading from data sheets. In addition, it prevents programming errors due to misreading of specifications, thereby reducing man-hours for debugging. This allows reduction of program creation costs that are responsible for increases in the cost and time preceding the start of data collection.

5. Conclusion

The author has demonstrated the feasibility and practicability of the idea of using a PLC as a data hub via programs automatically generated by the approach presented herein to convert raw data from various control devices, including analog sensors systematically into IoT data.

At present, there are only a limited number of devices registered in the Device Information Library. To cover many devices available in the market in the future, it will be necessary to develop a mechanism for managing and controlling the registration and updating of parameters.

This time, a relational database was selected as the destination of transmitted data. The author will deploy this to different types of IT connections to help develop a system for connecting information available from FA sites to various IT ecosystems.

References

1) Fujimoto, T. Competence Development Competition (in Japanese). Tokyo: Chuokoron-Shinsha, 2003, 406 p.
2) Ministry of Public Management, Home Affairs, Posts and Telecommunications, Ministry of Health, Labour and Welfare, and Ministry of Education, Culture, Sports, Science and Technology. FY2018 White Paper on Manufacturing Industries (Monodzukuri) (in Japanese). Tokyo: 2018, 325 p.
3) Omron Corporation. "Co-creative Onsite Data Service i-BELT: Omron Control Device" (in Japanese). September 20, 2018. https://www.fa.omron.co.jp/solution/i-belt/. (accessed 2018-12-19).
4) Industrie 4.0 Platform. Industry 4.0 Implementation Strategy (in

Japanese). 2015.
5) Special Committee for Smart Manufacturing. Manufacturing 2030 Ver. FY2016 (in Japanese). Tokyo: Japan Electrical Manufacturers' Association, 2016, 76 p.
6) Japan Switchboard & Control System Industries Association et al. Introduction to IEC61131−3 (in Japanese). Tokyo: PLCopen Japan, 2011, 105 p.

About the Author

KOBAYASHI Yoshiaki

Technology Dept. 1
Technology Development Division H.Q.
Industrial Automation Company
Speciality: Software engineering