YASKAWA Cockpit
Add-on Application Software
Programmer’s Manual

Upon receipt of the product and prior to initial operation, read these instructions thoroughly, and retain for future reference.

Please have the following information available when contacting Yaskawa Customer Support:
- System
- Primary Application
- Software Version (Located on Programming Pendant by selecting: [Main Menu] - [System Info] - [Version])
- Robot Serial Number (Located on robot data plate)
- Robot Sales Order Number (Located on controller data plate)

Part Number: 186634-1CD
Revision: 1
DANGER

- Read this manual carefully and be sure to understand its contents. Any matter, including operation, usage, measures, and an item to use, not described in this manual must be regarded as “prohibited” or “improper”.

NOTICE

- YASKAWA may modify this model without notice when necessary due to product improvements, modifications, or changes in specifications.
- If such modification is made, the manual number will also be revised.
- If your copy of the manual is damaged or lost, contact a YASKAWA representative to order a new copy. The representatives are listed on the back cover. Be sure to tell the representative the manual number listed on the front cover.

Registered Trademark

In this manual, names of companies, corporations, or products are trademarks, registered trademarks, or brand names for each company or corporation. The indications of (R) and ™ are omitted.
Notes for Safe Operation

Read this manual carefully before installation, operation, maintenance, or inspection of the YRC1000micro/YRC1000/DX200/DX100/RM100.

In this manual, the Notes for Safe Operation are classified as “DANGER”, “WARNING”, “CAUTION”, or “NOTICE”.

**DANGER**
Indicates an imminently hazardous situation which, if not avoided, will result in death or serious injury. Safety Signs identified by the signal word DANGER should be used sparingly and only for those situations presenting the most serious hazards.

**WARNING**
Indicates a potentially hazardous situation which, if not avoided, will result in death or serious injury. Hazards identified by the signal word WARNING present a lesser degree of risk of injury or death than those identified by the signal word DANGER.

**CAUTION**
Indicates a hazardous situation, which if not avoided, could result in minor or moderate injury. It may also be used without the safety alert symbol as an alternative to “NOTICE”.

**NOTICE**
NOTICE is the preferred signal word to address practices not related to personal injury. The safety alert symbol should not be used with this signal word. As an alternative to “NOTICE”, the word “CAUTION” without the safety alert symbol may be used to indicate a message not related to personal injury.

Even items described as “CAUTION” may result in a serious accident in some situations. At any rate, be sure to follow these important items.

**NOTE**
To ensure safe and efficient operation at all times, be sure to follow all instructions, even if not designated as “DANGER”, “WARNING” and “CAUTION”.
## Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction to YASKAWA Cockpit</td>
<td>1-1</td>
</tr>
<tr>
<td></td>
<td>1.1 Software Configuration</td>
<td>1-1</td>
</tr>
<tr>
<td></td>
<td>1.2 YCP Execution Environment</td>
<td>1-2</td>
</tr>
<tr>
<td></td>
<td>1.3 Logical Architecture</td>
<td>1-3</td>
</tr>
<tr>
<td>2</td>
<td>Registering Device Configurations</td>
<td>2-1</td>
</tr>
<tr>
<td></td>
<td>2.1 GUI</td>
<td>2-1</td>
</tr>
<tr>
<td></td>
<td>2.2 Web API</td>
<td>2-1</td>
</tr>
<tr>
<td></td>
<td>2.3 Device Configuration File</td>
<td>2-1</td>
</tr>
<tr>
<td>3</td>
<td>Data</td>
<td>3-1</td>
</tr>
<tr>
<td></td>
<td>3.1 Collected Data (Raw Data) and Analysis Results</td>
<td>3-1</td>
</tr>
<tr>
<td></td>
<td>3.2 Attributes</td>
<td>3-2</td>
</tr>
<tr>
<td></td>
<td>3.3 Files</td>
<td>3-3</td>
</tr>
<tr>
<td>4</td>
<td>Add-On Apps</td>
<td>4-1</td>
</tr>
<tr>
<td></td>
<td>4.1 Types</td>
<td>4-1</td>
</tr>
<tr>
<td></td>
<td>4.2 DA Apps (Data Analysis Application Software)</td>
<td>4-2</td>
</tr>
<tr>
<td></td>
<td>4.2.1 System Flow of a DA App</td>
<td>4-3</td>
</tr>
<tr>
<td></td>
<td>4.2.2 Adding a DA App</td>
<td>4-3</td>
</tr>
<tr>
<td></td>
<td>4.3 DC Apps (Data Collection Application Software)</td>
<td>4-4</td>
</tr>
<tr>
<td></td>
<td>4.3.1 System Flow of a DC App</td>
<td>4-5</td>
</tr>
<tr>
<td></td>
<td>4.3.2 Adding a DC App</td>
<td>4-5</td>
</tr>
<tr>
<td></td>
<td>4.4 DW App (Add-On GUI)</td>
<td>4-6</td>
</tr>
<tr>
<td></td>
<td>4.4.1 System Flow of a DW App</td>
<td>4-7</td>
</tr>
<tr>
<td></td>
<td>4.4.2 Adding a DW App</td>
<td>4-7</td>
</tr>
<tr>
<td></td>
<td>4.4.3 Version Management of DW Apps</td>
<td>4-7</td>
</tr>
<tr>
<td></td>
<td>4.4.4 Support for Multiple Languages</td>
<td>4-8</td>
</tr>
<tr>
<td></td>
<td>4.4.5 Important Information on DW Apps</td>
<td>4-9</td>
</tr>
<tr>
<td></td>
<td>4.5 Add-On App License Management</td>
<td>4-10</td>
</tr>
<tr>
<td></td>
<td>4.5.1 License Management through Web API</td>
<td>4-10</td>
</tr>
<tr>
<td></td>
<td>4.6 Development Environments for Add-On Apps</td>
<td>4-11</td>
</tr>
<tr>
<td></td>
<td>4.6.1 Important Information When Creating JSON Files</td>
<td>4-11</td>
</tr>
<tr>
<td>5</td>
<td>Web API</td>
<td>5-1</td>
</tr>
</tbody>
</table>
Contents

5.1 Web API Calls and Responses........................................................................................................... 5-1
  5.1.1 Web API Calls ......................................................................................................................... 5-1
  5.1.2 Web API Responses ................................................................................................................... 5-2
5.2 Web API Groups..................................................................................................................................... 5-3
  5.2.1 User Authentication .................................................................................................................... 5-3
  5.2.2 License Validation ......................................................................................................................... 5-4
  5.2.3 Register, Retrieve, and Delete Data Items that Handle Data ..................................................... 5-4
  5.2.4 Store, Retrieve, and Delete Raw Data (Collected Data from Equipment) .................................. 5-5
  5.2.5 Store, Retrieve, and Delete Analysis Data .................................................................................. 5-6
  5.2.6 Register and Delete Attribute Information .................................................................................. 5-6
5.3 Query Function for Data Search Web API Calls .................................................................................... 5-7
  5.3.1 Item That Defines the Query Statement ....................................................................................... 5-7
  5.3.2 Specifications for the Query Statement ....................................................................................... 5-8
  5.3.3 Errors Related to the Query Statement ....................................................................................... 5-9
6 Security..................................................................................................................................................... 6-1
  6.1 User Authority Management for the Web API................................................................................. 6-1
    6.1.1 User Authority Management for DA Apps and DC Apps ......................................................... 6-1
    6.1.2 User Authority Management for DW Apps ................................................................................. 6-1
7 Log Output.................................................................................................................................................. 7-1
8 Registering Applications with YCP Resource Monitor ........................................................................... 8-1
9 Appendix.................................................................................................................................................. 9-1
  9.1 Naming Conventions in YCP ............................................................................................................ 9-1
  9.2 List of Specifications for User Input ................................................................................................ 9-1
  9.3 Application Names and Application IDs ......................................................................................... 9-4
  9.4 Sample Program............................................................................................................................... 9-5
    9.4.1 Overview ......................................................................................................................................... 9-5
    9.4.2 DA App ............................................................................................................................................ 9-6
    9.4.3 DW App ............................................................................................................................................ 9-11
1 Introduction to YASKAWA Cockpit

1.1 Software Configuration

YASKAWA Cockpit is software (called “YCP” in this manual) that implements a framework which allows for the addition of application softwares that collect and accumulate data from various types of controllers and application softwares that view and analyze that collected data. fig. 1-1 “Configuration of YCP” shows this software configuration. table 1-1 “Elements in the Software Configuration of YCP” gives descriptions of the elements in fig. 1-1.

Fig. 1-1: Configuration of YCP

![Configuration Diagram]

Table 1-1: Elements in the Software Configuration of YCP

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YASKAWA Cockpit</td>
<td>YCP provides the HTTP-based GUI and web API. It provides functions to register/delete configuration information and collect/store data through the GUI and web API.</td>
</tr>
<tr>
<td>Database</td>
<td>Data collected/stored through YCP is stored to the database. Using the web API provided by YCP, the database can be accessed by add-on application software (called “Add-on Apps” in this manual).</td>
</tr>
<tr>
<td>File system</td>
<td>The file system stores the static application software used by the HTTP-based GUI, such as .html and .css files.</td>
</tr>
</tbody>
</table>
1.2 YCP Execution Environment

The following table gives the execution environment of YCP. YCP runs in a Java VM, and it uses Apache Tomcat as the servlet container and Spring Framework as the framework.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data analysis application software (DA App)</td>
<td>Application software that performs data analysis. These Apps use data stored in the database and execute processing on it, such as statistical analysis and machine learning.</td>
</tr>
<tr>
<td>Data collection application software (DC App)</td>
<td>Application software that performs data collection. These Apps collect data from various controllers and send that data to the database through the web API.</td>
</tr>
<tr>
<td>Add-on GUI (DW App)</td>
<td>Application software that views data (serves as a data window). These Apps display data stored in the database and the execution results of DA Apps.</td>
</tr>
</tbody>
</table>

Table 1-2: Execution Environment

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating system</td>
<td>Windows 10 Professional 64-bit</td>
</tr>
<tr>
<td>Java execution environment</td>
<td>Java Runtime Environment 1.8</td>
</tr>
<tr>
<td>Web browser</td>
<td>Google Chrome, etc.</td>
</tr>
<tr>
<td>Number of CPU cores</td>
<td>4</td>
</tr>
<tr>
<td>CPU</td>
<td>Core i7</td>
</tr>
<tr>
<td>RAM</td>
<td>16 GB or higher</td>
</tr>
<tr>
<td>Hard disk</td>
<td>500 GB or higher (1 TB recommended)</td>
</tr>
</tbody>
</table>
1.3 Logical Architecture

The YCP database has the hierarchical structure shown in the following diagram.

*Fig. 1-2: Hierarchical Structure and Stored Data*
YCP has a factory, line, cell, controller, mechanism, device structure from the top level in that order. The following table gives a description of each level. Of these structural elements, controllers, mechanisms, and devices can store various types of data. Data that can be stored is collected data (raw data), analysis results, attributes, and files. Shared is also defined as an area that does not belong to a level and that is shared in YCP. Attributes and files can be stored in the Shared area.

Table 1-3: Description of Levels

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factory</td>
<td>Represents the entire factory. A factory consists of one or multiple lines. One factory can be handled with one instance of YCP.</td>
</tr>
<tr>
<td>Line</td>
<td>Represents a line in the factory. A line consists of one or multiple cells.</td>
</tr>
<tr>
<td>Cell</td>
<td>A cell consists of one or multiple controllers. Example: Machine tool or cell on a line</td>
</tr>
<tr>
<td>Controller</td>
<td>Represents equipment that controls a mechanism or device. Example: Robot controller and machine controller</td>
</tr>
<tr>
<td>Mechanism</td>
<td>Represents equipment that consists of multiple axes and sensors and that controls position and posture in Cartesian coordinates and two-dimensional planes. The mechanism can be omitted if the controller directly controls axes. Example: Vertically articulated robot, parallel-link type robot, and X-Y table</td>
</tr>
<tr>
<td>Device</td>
<td>Equipment that controls motor position and speed on the motor of one axis (both rotary and linear types). Device also includes various types of sensors. Example: SERVOPACK and motor, AC Drive and motor, torque sensor, and temperature sensor</td>
</tr>
<tr>
<td>Shared</td>
<td>An area that does not belong to the above levels. The Shared area represents a location to share data between the various Apps in YCP.</td>
</tr>
</tbody>
</table>
2 Registering Device Configurations

The various levels (e.g., factory, line, cell, controller, mechanism, and device) can be registered using the YCP GUI, web API, and device configuration files.

2.1 GUI

You can register a factory, line, cell, controller, mechanism, and device from the YCP GUI. Refer to the following manual for details on the GUls used in registering a factory, line, cell, controller, mechanism, and device.

- YASKAWA Cockpit INSTRUCTIONS
  (Manual No.: HW1485838)

2.2 Web API

Refer to the following manual for details on the web API.

  (Manual No.: HW1485817)

2.3 Device Configuration File

You can register controllers, mechanisms, devices, and their configurations using the device configuration file.
The following table gives the specifications of the device configuration file.

**Table 2-1: Device Configuration File**

<table>
<thead>
<tr>
<th>Elements</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag Name</td>
<td>Number of Elements</td>
</tr>
<tr>
<td>Configuration</td>
<td>1</td>
</tr>
<tr>
<td>Controller</td>
<td>1..*</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Mechanism</td>
<td>0..*</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Device</td>
<td>0..*</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Device</td>
<td>0..*</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following sample shows the contents of a device configuration file.

```xml
<?xml version="1.0"?>
<Configuration version="2.0">
  <Controller key="FS100-1" name="FS100(1)" type="FS100">
    <Mechanism key="MH6H-1" name="MH6H(1)" type="MH6H">
      <Device key="S" name="S axis" />
      <Device key="L" name="L axis" />
    </Mechanism>
    <Device key="EX1" name="EX axis 1" />
  </Controller>
</Configuration>
```
3 Data

In YCP, collected data (raw data), analysis results, attributes, and files can be associated with a controller, mechanism, device, or the Shared area and stored (the Shared area is for attributes and files only). Use the web API to store, delete, modify, and perform other operations on the various types of data. Refer to chapter 5 “Web API” for the web API.

3.1 Collected Data (Raw Data) and Analysis Results

The storage destination and web API calls used for collected data and analysis results data are different, but functionally these types of data are the same. This section describes collected data and analysis results without distinguishing between the two.

Collected data and analysis results from a controller, mechanism, or device can be stored to YCP. Collected data and analysis results data are composed of a timestamp and the various items of data in that timestamp.

Fig. 3-1: Collected Data and Analysis Results

The timestamp represents the elapsed time from January 1, 1970, at 0:00:00 (unit [ms]). Data items other than the timestamp (e.g., torque value and sensor value) can be freely defined by the user of YCP. The following table gives the information that can be defined for each data item.

Table 3-1: Settings for Data Items

<table>
<thead>
<tr>
<th>Item</th>
<th>Required/ Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key name</td>
<td>Required</td>
<td>The name used when accessing the item with the web API, etc., after the item is registered.</td>
</tr>
<tr>
<td>Display name</td>
<td>Optional</td>
<td>The name used for displaying the item in the UI, etc.</td>
</tr>
<tr>
<td>Data type</td>
<td>Optional</td>
<td>The data type of the data item. The following primitive types and array types can be used in YCP. Integer (64 bits) Floating point number (64 bits, IEEE 754) String Boolean value (True, False)</td>
</tr>
<tr>
<td>Unit</td>
<td>Optional</td>
<td>The unit of the data item. Example: mm, m/s, or Nm</td>
</tr>
</tbody>
</table>
Registration and deletion for the key name, display name, and data type information for the data item can be performed through the web API on a controller, mechanism, or device that has been registered in advance. YCP also issues an event when data item registration and deletion are executed using the web API.

### 3.2 Attributes

An attribute is an area for storing attribute values and parameters of controllers, mechanisms, and devices. Attributes are stored to controllers, mechanisms, and devices, and attribute groups can be defined to group together attributes.

Fig. 3-2: Definitions of Attribute Groups

The following table gives the items that can be defined for each attribute.

<table>
<thead>
<tr>
<th>Item</th>
<th>Required/Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key name</td>
<td>Required</td>
<td>The name used when accessing the attribute with the web API, etc., after the attribute is registered.</td>
</tr>
<tr>
<td>Display name</td>
<td>Optional</td>
<td>The name used for displaying the item in the UI, etc.</td>
</tr>
</tbody>
</table>
| Data type     | Optional          | The data type of the attribute. The following primitive types and array types can be used in YCP.  
|               |                   | • Integer (64 bits)  
|               |                   | • Floating point number (64 bits, IEEE 754)  
|               |                   | • String  
|               |                   | • Boolean value (True, False)                                                  |
| Unit          | Optional          | The unit of the attribute. Example: mm, m/s, or Nm                          |
Registration and deletion for an attribute and attribute group can be performed through the web API on a controller, mechanism, or device that has been registered in advance.

### 3.3 Files

Files can be associated with controllers, mechanisms, and devices that have been registered in advance and stored. File groups can also be defined to group together files.

*Fig. 3-3: Definitions of File Groups*

Registration and deletion for a file and file group can be performed through the web API on a controller, mechanism, or device that has been registered in advance. YCP also issues an event when file and file group registration and deletion are executed using the web API.
4 Add-On Apps

4.1 Types

There are three types of Add-on Apps.

Fig. 4-1: Types of Add-On Apps

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data analysis application software (DA App)</td>
<td>Application software that performs data analysis. These Apps use data stored in the database and execute processing on it, such as statistical analysis and machine learning.</td>
</tr>
<tr>
<td>Data collection application software (DC App)</td>
<td>Application software that performs data collection. These Apps collect data from various controllers and store that data to the database through the web API.</td>
</tr>
<tr>
<td>Add-on GUI (DW App)</td>
<td>Application software that views data (serves as a data window). These Apps display data stored in the database and the execution results of DA Apps.</td>
</tr>
</tbody>
</table>

DC Apps and DA Apps are implemented as processes that operate separately from the database, and they access the database in the YCP and various types of data through the web API.
4.2 DA Apps
(Data Analysis Application Software)

The following diagram shows the configuration of a DA App.

Fig. 4-2: Configuration of a DA App

A DA App takes data collected from controllers, mechanisms, and devices and analysis results, attributes, and files of other DA Apps as input, and it stores the results of executing various types of processing, such as data analysis and data processing, to the database. The App is implemented as a separate executable file from YCP.

Use the web API to read/write data to/from the database. For this reason, the App can be executed using either of the following methods:

• Execute the DA App on the computer that runs YCP.
• Execute the DA App on a separate device (e.g., computer) connected to the network.

A DA App can also have its own unique user interface.
### 4.2.1 System Flow of a DA App

The following diagram illustrates the flow of DA App execution processing in a simple manner.

![System Flow Diagram](image)

Refer to chapter 6 “Security” for user authentication. Refer to chapter 4.5 “Add-On App License Management” for the license check.

### 4.2.2 Adding a DA App

When adding an application, store it in the following folder. (The location may depend on the installation settings.)

```
C:\Program Files\YASKAWA\YASKAWA Cockpit\Addon\AddonApplicationSoftware Folder
```

Note: “AddonApplicationSoftware Folder” is a folder for a developer to create. Create a folder with the name of the application. Refer to chapter 9.3 “Application Names and Application IDs” for the application name.

YCP also has YCP Resource Monitor which monitors the CPU and memory usage of YCP.

When adding an application, register it with YCP Resource Monitor.

Refer to chapter 8 “Registering Applications with YCP Resource Monitor” for the registration method.
4.3 DC Apps (Data Collection Application Software)

The following diagram shows the configuration of a DC App.

Fig. 4-3: Configuration of a DC App

A DC App collects data related to controllers, mechanisms, and devices, and transmits the collected data to the database. The App is implemented as a separate executable file from YCP. To collect data from controllers, mechanisms, and devices, the DC App must be compatible with the communications medium and protocol for each controller, mechanism, and device.

Use the web API to store data to the database. For this reason, the App can be executed using either of the following methods:

- Execute the DC App on the computer that runs YCP.
- Execute the DC App on a separate device (e.g., computer) connected to the network.

A DC App can also have its own unique user interface.
4.3 DC Apps (Data Collection Application Software)

4.3.1 System Flow of a DC App

The following diagram illustrates the flow of DC App execution processing in a simple manner.

![System Flow Diagram]

Refer to *chapter 6 “Security”* for user authentication. Refer to *chapter 4.5 “Add-On App License Management”* for the license check.

4.3.2 Adding a DC App

When adding an application, store it in the following folder. (The location may depend on the installation settings.)

C:\Program Files\YASKAWA\YASKAWA Cockpit\Addon\ “AddonApplicationSoftware Folder”

Note: “AddonApplicationSoftware Folder” is a folder for a developer to create. Create a folder with the name of the application. Refer to *chapter 9.3 “Application Names and Application IDs”* for the application name.

YCP also has YCP Resource Monitor which monitors the CPU and memory usage of YCP.

When adding an application, register it with YCP Resource Monitor. Refer to *chapter 8 “Registering Applications with YCP Resource Monitor”* for the registration method.
4.4 DW App (Add-On GUI)

The following diagram shows the configuration of a DW App.

*Fig. 4-4: Configuration of a DW App*

A DW App can be associated with factories, lines, cells, or controllers to visualize data (collected data, analysis results, attributes, and files) stored in the database.

When a DW App is registered in YCP, it is stored to the file system. To use a DW App, access the YCP GUI from a web browser, and click the menu of the DW App from the page on the level at which the DW App has been associated. When the DW App is started, the App stored in storage is loaded, rendered, and executed in the web browser.

DW Apps can use the following features and documents supported as standard by the web browser.

- HTML documents
- Images (e.g., .jpg, .png, and .gif)
- Stylesheets (.css)
- JavaScript

You can create a GUI that accesses and modifies data (collected data, analysis results, attributes, and files) held in the database by calling the web API using JavaScript.
4.4.1 System Flow of a DW App

The following diagram illustrates the flow of DW App execution processing in a simple manner.

![System Flow Diagram]

Refer to chapter 4.5 “Add-On App License Management” for the license check.

4.4.2 Adding a DW App

A DW App is added from the YCP user interface.

Refer to the following manual for how to add the application from the user interface.

- YASKAWA Cockpit Add-On Application Software Development Manual (Manual No.: HW1485815)
  Chapter 4.3.2.3 "Registering DW Apps"

4.4.3 Version Management of DW Apps

Add version information to the DW App at registration so that the version of the App can be checked. Version management is not performed for DA Apps and DC Apps because registration is not required.

4.4.3.1 Manifest File

Provide the version information as a file. This file is called the manifest file. Add the manifest file to the DW App file (zip).

The following section gives the procedure for creating the manifest file.

**Table 4-2: Specifications of the Manifest File**

<table>
<thead>
<tr>
<th>Key name</th>
<th>Description</th>
<th>Usable Characters</th>
<th>Maximum Length</th>
<th>Required/Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddOn-Title</td>
<td>DW App name</td>
<td>The following characters cannot be used. \ / : * ? &lt; &gt;</td>
<td>= &amp;</td>
<td>1 to 32</td>
</tr>
<tr>
<td>AddOn-Version</td>
<td>DW version</td>
<td>Only alphanumeric characters and the following symbols can be used. \ / _ ^ *</td>
<td>1 to 32</td>
<td>Required</td>
</tr>
<tr>
<td>AddOn-Vendor</td>
<td>DW vendor name</td>
<td>The following characters cannot be used. \ / : * ? &lt; &gt;</td>
<td>= &amp;</td>
<td>1 to 256</td>
</tr>
</tbody>
</table>

Note: The character encoding is UTF-8.
4.4.3.2 Checking Version Information

Version information is displayed on the initial settings add-on page. The GUI is shown below.

Fig. 4-5: Version Information GUI

4.4.3.3 Version Information Creation Tool

DW apps created by a third-party vendor other than YASKAWA also require a manifest file. YCP provides a tool (Manifest File Tool) that supports the creation of manifest files. You can find the tool on the installation DVD.

Refer to the following manual for the manifest file creation procedure.

- YASKAWA Cockpit Add-On Application Software Development Manual (Manual No.: HW1485815)
  Chapter 4.3.2.2 "Manifest File Creation Procedure"

4.4.4 Support for Multiple Languages

By creating language files, a DW App can change to a view that corresponds to the language selected on the YCP login window.

A language file must be created and added to the DW App file (zip) for each language that is supported.

The following table gives the specifications of the language file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File Name</td>
<td>message_&lt;YCP-supported language code1&gt;.json</td>
</tr>
<tr>
<td>Message Notation</td>
<td>JSON format (key: value)</td>
</tr>
<tr>
<td>File Format</td>
<td>UTF-8</td>
</tr>
</tbody>
</table>

1 English and Japanese are supported by default.

The YCP-supported language code for Japanese and English are as follows:
Japanese: jp
English: en
4 Add-On Apps
4.4 DW App (Add-On GUI)

4.4.4.1 Loading Language Files

When a DW App file (zip) is registered on the YCP web view, the language files are stored in the following folders.

C:\ProgramFiles\YASKAWA\YASKAWA\Cockpit\Tomcat8.5\webapps\YCP\App\resources\json_messages\"addonUIName"

"addonUIName" is the UI name that is set when the DW App is registered on the YCP web view.

The stored language files are loaded by the script (e.g., JavaScript) and displayed on the window.

When loading a language file, specify the file name using JSP Expression Language (EL) expression.

An example is given next.

Example: To specify the location of the message file in the JSP file using an EL expression:

```
${addOnUIName}: Add-on UI name
${locale}: Locale
${contextPath}: Context path
```

The following figure shows an overview of the processing from when the DW App gets the language file to when it is displayed on the window.

Fig. 4-6: Overview of the Processing to Display the Language during DW App Execution

4.4.5 Important Information on DW Apps

DW Apps are executed in the web browser. For this reason, the source code of the DW App can be easily viewed and accessed if you use the view page source function in the web browser. Therefore, do not include any processing that will be problematic if the information in the DW App is leaked, such as proprietary information or algorithms.
4.5 Add-On App License Management

YCP uses license files to manage add-on licenses.

Refer to the following manual for information on the license file transaction.

- YASKAWA Cockpit Add-On Application Software Development Manual (Manual No.: HW1485815)
  Chapter 4.4 “Add-On App License Management”

4.5.1 License Management through Web API

Add-on Apps can retrieve the following information through the web API.

- The license is valid?
- License expiration date

Whether or not to start/execute a Add-on App can be determined by the result of the web API call. This determination is left to the provider and implementer of the App.

Refer to the following manual for details on the web API.

- YASKAWA Cockpit Add-On Application Software Web API Reference Manual (Manual No.: HW1485817)

Fig. 4-7: License Check API Call
4.6 Development Environments for Add-On Apps

The following table gives the development environments for Add-on Apps.

<table>
<thead>
<tr>
<th>App Type</th>
<th>Development Environment</th>
</tr>
</thead>
</table>
| DA App   | The following functions can be implemented:  
• Issue HTTP request and analyze HTTP response  
• Serialize/deserialize JSON  
Example: Visual C++/C#, Eclipse (Java) |
| DC App   | The following functions can be implemented:  
• Issue HTTP request and analyze HTTP response  
• Serialize/deserialize JSON  
• Capable of communicating with the controller for data collection (supports the physical layer and protocol).  
Example: Visual C++/C#, Eclipse (Java) |
| DW App   | • Debugging environment contained in the web browser, such as Internet Explorer and Google Chrome.  
• Eclipse+JSDT plug-in.  
| Web browser | Debugging environment |
| Internet Explorer | F12 developer's tool  
Start with [F12] |
| Google Chrome | Chrome DevTools  
Start with [Ctrl] + [Shift] + [I] |

4.6.1 Important Information When Creating JSON Files

Use the standard Java or C# library*1 to create JSON files.

If the library is not used, do not use octal or hexadecimal notation for numeric values.

*1 YASKAWA recommends the following libraries as standard libraries.  
• C#: Json.NET (Newtonsoft)  
• Java: org.json
5 Web API

Refer to the following manual for details on the web API.
• YASKAWA Cockpit Add-On Application Software Web API Reference Manual (Manual No.: HW1485817)

5.1 Web API Calls and Responses

The web API provided by YCP is called from the client using HTTP. This section describes how to call the web API and the responses returned from the web API.

5.1.1 Web API Calls

Web API calls conform to HTTP. The following table gives the specifications.

Table 5-1: Specifications of Web API Calls

<table>
<thead>
<tr>
<th>Item</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port number</td>
<td>The default values are listed below.</td>
</tr>
<tr>
<td>Methods</td>
<td>GET, POST, PUT, DELETE</td>
</tr>
<tr>
<td>Request header</td>
<td>The following header is required.</td>
</tr>
<tr>
<td>Character encoding</td>
<td>UTF-8</td>
</tr>
<tr>
<td>HTTP version</td>
<td>HTTP1.1</td>
</tr>
</tbody>
</table>

The following is an example of a web API call (request).

```
POST/user/auth
Content-Type : application/json

{""Password":"admin", "UserName":"admin"}
```

Use URL encoding for characters if parameters specified when making the call contain characters that cannot be used in a URL/URI, such as a single-byte space (%20 when encoded).
5.1.2 Web API Responses

Web API responses are returned in conformance with HTTP specifications. The character encoding is UTF-8. The following is an example of a response.

- **Input**
  - GET /factories

- **Response**
  - Response 200
  
  ```
  
  Factory: ["factory1", "factory2"]
  
  ```

Web API responses include an HTTP status code. Refer to RFC7231 (Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content) for details on HTTP status codes.

http://www.rfc-base.org/rfc-7231.html

<table>
<thead>
<tr>
<th>Table 5-2: Details of Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Successful</strong></td>
</tr>
<tr>
<td>HTTP status</td>
</tr>
<tr>
<td>200 OK</td>
</tr>
<tr>
<td>3xx Redirection</td>
</tr>
<tr>
<td>Response body</td>
</tr>
<tr>
<td>Web API content.</td>
</tr>
<tr>
<td><strong>Error</strong></td>
</tr>
<tr>
<td>HTTP status</td>
</tr>
<tr>
<td>4xx Client (web API caller) error</td>
</tr>
<tr>
<td>5xx Server (analysis module internal) error</td>
</tr>
<tr>
<td>Response body</td>
</tr>
<tr>
<td>Error message</td>
</tr>
<tr>
<td>The error message is not intended for the user of the add-on, but for the developer of the add-on and the developer of the analysis module.</td>
</tr>
</tbody>
</table>
5.2 Web API Groups

YCP has over 100 types of web API calls, including those for accessing data in the database.

The web API calls have been broadly categorized into the following groups:

- User authentication
- License validation
- Register, retrieve, and delete data items that handle data
- Store, retrieve, and delete raw data (collected data from equipment)
- Store, Retrieve, and Delete Analysis Data
- Register and delete attribute information

The following section describes an overview of each of these web API groups.

Refer to the following manual for details on the web API calls used in the examples and other web API calls.

  (Manual No.: HW1485817)

5.2.1 User Authentication

This group retrieves tokens. Refer to chapter 6 “Security” for tokens.

Example: Request Content

```
POST /users/auth
```

Add the user name and password to the content portion of the request. When the user name and password are authenticated, a token and token expiry date are returned.

Example: Request Content

```
{
  "UserName": "admin",
  "Password": "admin"
}
```

Example: Response

```
{
  "Token": "abcdefghij0123456789",
  "Lifetime": 86400000
}
```
5.2.2 License Validation

This group validates application licenses. Refer to chapter 4.5 “Add-On App License Management” for details on licenses.

```
GET /apps/{app-key}/validate
```

Note: Specify the application name for `{app-key}`.

If the web API processing is successful, the validation result and license expiration date are returned.

Example: Response

```
{
  "Result": true,
  "ExpiryDate": "2018-12-31"
}
```

5.2.3 Register, Retrieve, and Delete Data Items that Handle Data

This group registers, retrieves, and deletes data items of data to store in the database.

This example shows the registration of item settings in the analysis data of a device.

```
POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/{controller-key}/mechanisms/{mechanism-key}/devices/{device-key}/apps/{app-key}/analysis/items
```

Note: Specify the application ID for `{app-key}`.

Add the item to register to the content portion of the request.

Example: Request Content (Register the "Torque" Item)

```
[
  {
    "Key": "Trq",
    "Type": "Double",
    "Unit": "Nm",
    "Name": "Torque",
    "Visible": false
  }
]
```

Item settings for raw data of a device are also registered as follows:

```
POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/{controller-key}/mechanisms/{mechanism-key}/devices/{device-key}/data/items
```

The format for the content portion of the request is the same as that for analysis data.
5.2.4 Store, Retrieve, and Delete Raw Data (Collected Data from Equipment)

This group can store, retrieve, and delete data in the database that was collected from equipment by a DC App.

This example shows the registration of raw data of a device.

Add the conditions for the data to retrieve to the content portion of the request. The conditions can be specified for each data item by including a query statement in the content request. Refer to chapter 5.3 “Query Function for Data Search Web API Calls” for queries.

The data is returned in JSON format according to the content.

Example: Request Content

```
{
  "Item" : ["TimeStamp", "JobNo", "Torque", "FbVel", "CmdPos"],
  "N" : 1000000,
  "Size" : 104857600,
  "Timeout" : 2147483647,
  "Query" : "(Torque > 14.5) AND (FbVel < 123.4) AND (CmdPos IS NOTNULL)",
  "Sort" : [
    {"Target" : "TimeStamp", "Ascending" : true},
    {"Target" : "JobNo", "Ascending" : false}
  ]
}
```

Example: Response

```
{
  "Header" : {
    "QueryResult" : 3000000,
    "Returned" : 1000000,
    "ReturnedTotal" : 2000000,
    "QueryResultChanged" : false,
    "ContinuationPoint" : "CKaEL"
  },
  "Data" : [
    {"TimeStamp" : 1501032052000 , "DataA" : 10.1 , "DataB" : 20 },
    {"TimeStamp" : 1501032052010 , "DataA" : 20.2 , "DataB" : 40 },
    {"TimeStamp" : 1501032052020 , "DataA" : 10.1 , "DataB" : 20 }
  ]
}
```

Collected data is also stored as follows:

```
POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/{controller-key}/mechanisms/{mechanism-key}/devices/{device-key}/data
```

Add the data to store to the content of the request. When storing collected data to a mechanism and device, add the data to the content portion of the request.

Example: Request Content

```
[
  {"TimeStamp":1501032052000, "Mechanism":"MH6H-1", "Device":"Axis-S","Value1":13.1, "Value2":[20,21]},
]
```
5.2.5 Store, Retrieve, and Delete Analysis Data

This group can store, retrieve, and delete analysis data in the database that was analyzed by the DA App.

This example shows the registration of analysis data of a device.

```
POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/
     {controller-key}/mechanisms/{mechanism-key}/devices/{device-key}/apps/
     {app-key}/analysis/search
```

Note: Specify the application ID for {app-key}.

The format for the content portion of the request is the same as that when storing raw data.

Analyzed data is also stored as follows:

```
POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/
     {controller-key}/apps/{app-key}/analysis
```

Note: Specify the application ID for {app-key}.

The format for the content portion of the request is the same as that when storing raw data.

5.2.6 Register and Delete Attribute Information

This group registers and deletes attribute information held by controllers, mechanisms, and devices.

This example shows how to set common attributes in an attribute group.

```
PUT /shared/attribute-groups/{group-key}/attributes
```

Add the common attribute settings to the content of the request.

Example: Request Content (Register the "Threshold" Attribute)

```
[
  {
    "Key":"Threshold",
    "Value":1234.5678,
    "Type":"Double",
    "Unit":"mm/s",
    "Name":"A threshold",
    "Visible":true
  }
]
```
5.3 Query Function for Data Search Web API Calls

For a portion of web API calls in YCP, conditions are specified for each data item by including a query statement in the request. Some web API calls also have a function that extracts data that matches a specified condition from the database. At present, the following web API calls have query functions.

<table>
<thead>
<tr>
<th>Web API</th>
<th>Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/{controller-key}/data/search</td>
<td>Retrieves collected data associated with the controller.</td>
</tr>
<tr>
<td>POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/{controller-key}/mechanisms/{mechanism-key}/data/search</td>
<td>Retrieves collected data associated with the mechanism.</td>
</tr>
<tr>
<td>POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/{controller-key}/devices/{device-key}/data/search</td>
<td>Retrieves collected data associated with the device. (When the device is directly under a controller.)</td>
</tr>
<tr>
<td>POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/{controller-key}/mechanisms/{mechanism-key}/devices/{device-key}/data/search</td>
<td>Retrieves collected data associated with the device.</td>
</tr>
<tr>
<td>POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/{controller-key}/apps/{app-key}/analysis/search</td>
<td>Retrieves analysis results associated with the controller.</td>
</tr>
<tr>
<td>POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/{controller-key}/mechanisms/{mechanism-key}/apps/{app-key}/analysis/search</td>
<td>Retrieves analysis results associated with the mechanism.</td>
</tr>
<tr>
<td>POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/{controller-key}/devices/{device-key}/apps/{app-key}/analysis/search</td>
<td>Retrieves analysis results associated with the device. (When the device is directly under a controller.)</td>
</tr>
<tr>
<td>POST /factories/{factory-key}/lines/{line-key}/cells/{cell-key}/controllers/{controller-key}/mechanisms/{mechanism-key}/devices/{device-key}/apps/{app-key}/analysis/search</td>
<td>Retrieves analysis results associated with the device.</td>
</tr>
<tr>
<td>POST /factories/{factory-key}/events/search</td>
<td>Retrieves event information.</td>
</tr>
</tbody>
</table>

5.3.1 Item That Defines the Query Statement

For the query statement, add the "query" item in the request. The "query" item is optional. The data for the full period stored in the database can be extracted by omitting this item.
5.3.2 Specifications for the Query Statement

The following information shows the rules of the query statement.

```
<query> ::= <condition>
<condition> ::= <condition> "AND" <condition>
| <condition> "OR" <condition>
| "(" <condition> ")"
| <item> <operator> <constant>
| <item> "EXISTS" <boolean>
<item> ::= item-key
<constant> ::= constant (integer, double, string, boolean)
<boolean> ::= "true" | "false"
```

Note: If a string value is specified as <constant>, the double quotation marks (") that enclose the string value for <constant> must be escaped using backslashes (\).

The following table gives the role of <operator>, "EXISTS", "AND", and "OR" listed in the query statement rules above.

**Table 5-3: Description of the Operators**

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>Equal to</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>Not equal to</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less than</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater than or equal to</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or equal to</td>
</tr>
<tr>
<td>EXISTS</td>
<td>The specified data item exists</td>
</tr>
<tr>
<td>AND</td>
<td>Two conditions are combined and both conditions are true</td>
</tr>
<tr>
<td>OR</td>
<td>Either of two conditions is true</td>
</tr>
</tbody>
</table>

The following table gives the precedence and evaluation direction (=associativity) of each operator. Operators that are higher in the table have higher precedence.

**Table 5-4: Operator Precedence and Associativity**

<table>
<thead>
<tr>
<th>Precedence</th>
<th>Operator</th>
<th>Associativity</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>=, &lt;&gt;, &gt;, &lt;, &gt;=, &lt;=, EXISTS</td>
<td>-</td>
</tr>
<tr>
<td>Low</td>
<td>AND</td>
<td>Left to right</td>
</tr>
<tr>
<td></td>
<td>OR</td>
<td>Left to right</td>
</tr>
</tbody>
</table>

However, if <condition> is enclosed in parentheses, the judgment of the portion in parentheses is given precedence.
5.3.3 Errors Related to the Query Statement

The web API returns errors related to the query statement in the following cases:

- If the query statement does not conform to the rules described in Specifications for the Query Statement and Specifications for the Query Statement.

- If an unsupported condition is specified for "<item> <operator> <constant>" in the rules.
  - The data type for the data item specified in <item> is not the same as the data type of the value specified for <constant> (example: a boolean data item is compared to an integer value).
  - An array type data item was specified for <item> (array type data cannot be specified as a condition).
6 Security

6.1 User Authority Management for the Web API

The web API client requires authentication information (tokens) when making web API calls.

6.1.1 User Authority Management for DA Apps and DC Apps

For DA Apps and DC Apps, use the web API calls for user authentication to get a token.

First, the web API client sends the user name and password using the web API call for user authentication.

When the user name and password are authenticated, a token is returned. The web API client calls the web API together with this retrieved token.

Conform to the following specifications when calling the web API.

- Type: apiKey
- Name: YCP-API-TOKEN
- Type: HEADER

6.1.2 User Authority Management for DW Apps

For DW Apps, a token can be used that is created based on the information when the user logged into YCP with the web browser.

This token has the same specification as the token that can be retrieved by using the user authentication web API.

When passing the token from YCP to the DW App, use the JSP Expression Language (EL) expression (${ycpDWToken}).

The DW App retrieves the token from the JSP EL expression and calls the web API using that token.

Conform to the following specifications when calling the web API.

- Type: apiKey
- Name: YCP-DW-TOKEN
- Type: HEADER

The following figure shows an overview diagram of the processing.

Fig. 6-1: Processing Overview
7 Log Output

This chapter gives the specifications for outputting the log.

Errors at the ERROR and WARNING levels are monitored by YCP Resource Monitor and displayed to the user.

Table 7-1: Log Output Specifications

<table>
<thead>
<tr>
<th>Item</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log file location</td>
<td>C:\Program Files\YASKAWA\YASKAWA Cockpit\APData\logs\Addon\ &quot;AddonApplicationSoftwareFolder&quot; Note: For &quot;AddonApplicationSoftware Folder&quot;, create a folder with the name of the application. Refer to chapter 9.3 “Application Names and Application IDs” for the application name.</td>
</tr>
<tr>
<td>File name</td>
<td>&quot;Application name&quot;_yyyyMMdd.log</td>
</tr>
<tr>
<td>Log Rotation</td>
<td>Once a day</td>
</tr>
<tr>
<td>Number of days to store in the log</td>
<td>30 days</td>
</tr>
<tr>
<td>Log format</td>
<td>yyyy-MM-dd HH:mm:ss.m’m’m’ level [className functionName threadNumber] errorCode=zzzzzz message=&quot;xxxxxxxx&quot;</td>
</tr>
</tbody>
</table>
  • yyyy-MM-dd: Year-Month-Day  
  • HH:mm:ss.m’m’m’: Hours-Minutes-Seconds.Milliseconds  
  • level: Log level (ERROR, WARN, INFO)  
  • className: Class name that output the log  
  • functionName: Method name that output the log  
  • threadNumber: Thread task number that output the log  
  • message: Log message  
  • errorCode: Error code (six digits) Note: Not necessary for INFO and WARN. |
| Differences in log levels | Refer to table 7-2 “Differences in Log Levels” |

Table 7-2: Differences in Log Levels

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERROR</td>
<td>Errors that include when YCP cannot continue to function and when the processing of a request cannot be continued. Example: If an OutOfMemoryError occurs.</td>
</tr>
<tr>
<td>WARN</td>
<td>Warnings that include when a web API parameter is incorrect and when the caller should correct the request. Example: The web API controller key is not found.</td>
</tr>
<tr>
<td>INFO</td>
<td>Information that includes the start and end of processing and dumps of processing time.</td>
</tr>
<tr>
<td>DEBUG</td>
<td>Information used only during debugging. This level includes information in more detail than INFO.</td>
</tr>
<tr>
<td>TRACE</td>
<td>Information used only during debugging. A log with too much information and that will impact the performance of YCP is output at the TRACE level.</td>
</tr>
</tbody>
</table>
8 Registering Applications with YCP Resource Monitor

Alarms and resource usage in DA Apps and DC Apps can be monitored with YCP Resource Monitor.

Refer to the following manual for YCP Resource Monitor.

- YASKAWA Cockpit Maintenance Manual
  (Manual No.: HW1485840)

Edit the following file when registering DA Apps and DC Apps with YCP Resource Monitor.
(The location of the file may depend on the installation settings.)

C:\Program Files\YASKAWA\YASKAWA Cockpit\conf\AddonTargetConfiguration.xml

Restart YCP after the file is edited.

Unedited File (Default File When YCP Is Installed)

```xml
<?xml version="1.0" encoding="utf-8"?>
<Configuration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns="http://www.cpandl.com">
    <Modules>
        <ModuleType Name="ResourceMonitor">
            <Modes Name="CPU"/>
            <Modes Name="MEMORY"/>
        </ModuleType>
        <ModuleType Name="AlarmMonitor">
            <Modes Name="Alarm"/>
        </ModuleType>
    </Modules>
</Configuration>
```
Refer to chapter 9.3 “Application Names and Application IDs” for the application name.
9 Appendix

9.1 Naming Conventions in YCP

Table 9-1: Naming Conventions

<table>
<thead>
<tr>
<th>Item</th>
<th>Rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web API URI (Including path parameter)</td>
<td>Use all lowercase characters and connect words with a hyphen (chain case).</td>
</tr>
<tr>
<td>Body of web API response/request (JSON)</td>
<td>Start words with capital letters and connect the words together (camel case).</td>
</tr>
<tr>
<td>Device configuration file (XML)</td>
<td>For tag names, start words with capital letters and connect the words together (camel case). For attribute names, use all lowercase characters and connect words with a hyphen (chain case).</td>
</tr>
<tr>
<td>Event names</td>
<td>'YE_' + &lt;Object&gt; + 'Event'. For &lt;Object&gt;, start words with capital letters and connect the words together (camel case).</td>
</tr>
</tbody>
</table>

9.2 List of Specifications for User Input

There are many types of input values and setting values in YCP. The following specifications are applied to all of these input values and setting values.

1. Strings that begin with the following character combinations are reserved for the system and cannot be used by the user.

2. Strings that begin with “_” cannot be used.

3. The character encoding is UTF-8.

The following table gives specifications for individual input values and setting values.

Table 9-2: List of Specifications for User Input

<table>
<thead>
<tr>
<th>Item</th>
<th>Input Destination</th>
<th>Usable Characters</th>
<th>Maximum Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factory name (display name)</td>
<td>GUI</td>
<td>Chars except for .:*?&lt;&gt;</td>
<td>=&amp;&quot;</td>
</tr>
<tr>
<td>Factory name (database name)</td>
<td>GUI</td>
<td>a-zA-Z0-9_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Line name (display name)</td>
<td>GUI</td>
<td>Chars except for .:*?&lt;&gt;</td>
<td>=&amp;&quot;</td>
</tr>
<tr>
<td>Line name (database name)</td>
<td>GUI</td>
<td>a-zA-Z0-9_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Cell name (display name)</td>
<td>GUI</td>
<td>Chars except for .:*?&lt;&gt;</td>
<td>=&amp;&quot;</td>
</tr>
<tr>
<td>Cell name (database name)</td>
<td>GUI</td>
<td>a-zA-Z0-9_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Controller name (display name)</td>
<td>GUI</td>
<td>Chars except for .:*?&lt;&gt;</td>
<td>=&amp;&quot;</td>
</tr>
<tr>
<td>Controller name (database name)</td>
<td>GUI</td>
<td>a-zA-Z0-9_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Controller serial no</td>
<td>GUI</td>
<td>Chars except for .:*?&lt;&gt;</td>
<td>=&amp;&quot;</td>
</tr>
<tr>
<td>Controller IP address</td>
<td>GUI</td>
<td>NNN.NNN.NNN.NNN (NNN : 0 to 255)</td>
<td></td>
</tr>
<tr>
<td>Controller D variable</td>
<td>GUI</td>
<td>0 to 1999</td>
<td></td>
</tr>
<tr>
<td>Controller name (key)</td>
<td>configuration file(.xml)</td>
<td>a-zA-Z0-9_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Controller name (name)</td>
<td>configuration file(.xml)</td>
<td>Chars except for .:*?&lt;&gt;</td>
<td>=&amp;&quot;, charset: UTF-8</td>
</tr>
<tr>
<td>Controller type (type)</td>
<td>configuration file(.xml)</td>
<td>a-zA-Z0-9_- /^_-</td>
<td>1 to 32</td>
</tr>
<tr>
<td>Mechanism name (key)</td>
<td>configuration file(.xml)</td>
<td>a-zA-Z0-9_-</td>
<td>1 to 16</td>
</tr>
</tbody>
</table>
9 Appendix

9.2 List of Specifications for User Input

<table>
<thead>
<tr>
<th>Item</th>
<th>Input Destination</th>
<th>Usable Characters</th>
<th>Maximum Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mechanism name (name)</td>
<td>configuration file(.xml)</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;, charset: UTF-8</td>
</tr>
<tr>
<td>Mechanism type (type)</td>
<td>configuration file(.xml)</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 32</td>
</tr>
<tr>
<td>Device name (key)</td>
<td>configuration file(.xml)</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Device name (name)</td>
<td>configuration file(.xml)</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;, charset: UTF-8</td>
</tr>
<tr>
<td>Device name (type)</td>
<td>configuration file(.xml)</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 32</td>
</tr>
<tr>
<td>Item name (key)</td>
<td>Web API</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Item name (name)</td>
<td>Web API</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;, charset: UTF-8</td>
</tr>
<tr>
<td>Item type (type)</td>
<td>Web API</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 32</td>
</tr>
<tr>
<td>Item unit (unit)</td>
<td>Web API</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Item value</td>
<td>Web API</td>
<td>Follows JSON specification, charset: UTF-8</td>
<td>0 to 256</td>
</tr>
<tr>
<td>DW application name (key)</td>
<td>GUI</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>DW application description</td>
<td>GUI</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;</td>
</tr>
<tr>
<td>DC application name (key)</td>
<td>Web API</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>DC application description</td>
<td>Web API</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;, charset: UTF-8</td>
</tr>
<tr>
<td>Attribute group name (key)</td>
<td>Web API</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Attribute key</td>
<td>Web API</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Attribute name</td>
<td>Web API</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;, charset: UTF-8</td>
</tr>
<tr>
<td>Attribute type</td>
<td>Web API</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 32</td>
</tr>
<tr>
<td>Attribute unit</td>
<td>Web API</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Attribute value</td>
<td>Web API</td>
<td>Follows JSON specification, charset: UTF-8</td>
<td>0 to 256</td>
</tr>
<tr>
<td>File group name (key)</td>
<td>Web API</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 255</td>
</tr>
<tr>
<td>Filename (key)</td>
<td>Web API</td>
<td>Full-width chars (alphanumeric, katakana, hiragana, Chinese character)</td>
<td>1 to 255</td>
</tr>
<tr>
<td>User first name</td>
<td>GUI</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;</td>
</tr>
<tr>
<td>User last name</td>
<td>GUI</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;</td>
</tr>
<tr>
<td>User login name</td>
<td>GUI</td>
<td>a-zA-Z0-9 .^_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>User password</td>
<td>GUI</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;</td>
</tr>
<tr>
<td>User e-mail address</td>
<td>GUI</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;</td>
</tr>
<tr>
<td>User Phone number</td>
<td>GUI</td>
<td>0 to 9</td>
<td>1 to 13</td>
</tr>
<tr>
<td>Mail host name (SMTP)</td>
<td>GUI</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;</td>
</tr>
<tr>
<td>Mail port number (SMTP)</td>
<td>GUI</td>
<td>1000 to 65535</td>
<td>1 to 256</td>
</tr>
<tr>
<td>Mail user name (SMTP)</td>
<td>GUI</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;</td>
</tr>
<tr>
<td>Mail user password (SMTP)</td>
<td>GUI</td>
<td>Chars except for V:=&quot;&lt;&gt;</td>
<td>=&quot;</td>
</tr>
<tr>
<td>ActiveMQ port number</td>
<td>GUI</td>
<td>1000 to 65535</td>
<td>1 to 256</td>
</tr>
<tr>
<td>ActiveMQ IP address</td>
<td>GUI</td>
<td>NNN.NNN.NNN.NNN (NNN : 0 to 255)</td>
<td></td>
</tr>
</tbody>
</table>
### Table 9-2: List of Specifications for User Input

<table>
<thead>
<tr>
<th>Item</th>
<th>Input Destination</th>
<th>Usable Characters</th>
<th>Maximum Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-mail title (notification)</td>
<td>Web API</td>
<td>All chars, charset: UTF-8</td>
<td>0 to 256</td>
</tr>
<tr>
<td>E-mail body (notification)</td>
<td>Web API</td>
<td>All chars, charset: UTF-8</td>
<td>1 to 65536</td>
</tr>
<tr>
<td>Event type (topic name)</td>
<td>Web API</td>
<td>a-zA-Z0-9-_</td>
<td>1 to 256</td>
</tr>
<tr>
<td>Event body (topic body)</td>
<td>Web API</td>
<td>All ASCII chars (Multi-byte chars cannot be used.)</td>
<td>1 to 65536</td>
</tr>
<tr>
<td>DA application name (key)</td>
<td>Web API</td>
<td>a-zA-Z0-9_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>DA application description</td>
<td>Web API</td>
<td>Chars except for .:*?&lt;&gt;</td>
<td>=&amp;&quot;, charset: UTF-8</td>
</tr>
<tr>
<td>DW Parameter key</td>
<td>GUI, DW property</td>
<td>a-zA-Z0-9_-</td>
<td>1 to 16</td>
</tr>
<tr>
<td>DW Parameter name</td>
<td>GUI, DW property</td>
<td>Chars except for .:*?&lt;&gt;</td>
<td>=&amp;&quot;, charset: UTF-8</td>
</tr>
<tr>
<td>DW Parameter type</td>
<td>GUI, DW property</td>
<td>a-zA-Z0-9-./^_-</td>
<td>1 to 32</td>
</tr>
<tr>
<td>DW Parameter value</td>
<td>GUI, DW property</td>
<td>Follows JSON specification, charset: UTF-8</td>
<td>1 to 256</td>
</tr>
<tr>
<td>Export name</td>
<td>GUI</td>
<td>Chars except for .:*?&lt;&gt;</td>
<td>=&amp;&quot;, charset: UTF-8</td>
</tr>
<tr>
<td>Export path</td>
<td>GUI</td>
<td>Chars except for .:*?&lt;&gt;</td>
<td>=&amp;&quot;, charset: UTF-8</td>
</tr>
<tr>
<td>Export server</td>
<td>GUI</td>
<td>NNN.NNN.NNN.NNN (NNN : 0 to 255)</td>
<td></td>
</tr>
</tbody>
</table>
9.3 Application Names and Application IDs

Application names and application IDs have been determined in advance in order to centrally manage add-on applications and maintain the consistency of data in the database.

Application names are defined as a maximum of 32 characters and application IDs are defined as 16 numbers.

The application names and application IDs are also used to identify application resources and alarms on the computer and to perform maintenance.

The application name and application ID are used in the following locations:

<table>
<thead>
<tr>
<th>DA App and DC App</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution module name</td>
<td>Application name</td>
</tr>
<tr>
<td>ID name in database (AppKey)</td>
<td>Application ID</td>
</tr>
<tr>
<td>License name</td>
<td>Application name</td>
</tr>
<tr>
<td>Log storage folder name</td>
<td>Application name</td>
</tr>
<tr>
<td>Log file name</td>
<td>Application name</td>
</tr>
<tr>
<td>Application temporary file storage folder name</td>
<td>Application name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DW App</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>DWzip name</td>
<td>Application name</td>
</tr>
<tr>
<td>Application name in MANIFEST file</td>
<td>Application name</td>
</tr>
<tr>
<td>ID name in database (AppKey)</td>
<td>Application ID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DA App and DC App Installer</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Shortcut name</td>
<td>Application name</td>
</tr>
<tr>
<td>Application storage folder name</td>
<td>Application name</td>
</tr>
<tr>
<td>Log storage folder name</td>
<td>Application name</td>
</tr>
<tr>
<td>Resource monitor registration name</td>
<td>Application name</td>
</tr>
</tbody>
</table>
9.4 Sample Program

9.4.1 Overview

There is a sample program for both a DA App and DC App.

The DA App retrieves torque values stored in the database and analyzes the maximum torque value from the retrieved values. The analyzed results are also stored in the database.

The DW App retrieves the torque values and maximum torque values stored in the database and displays those on the YCP web view as a graph.

9.4.1.1 Before Executing the Sample Programs

1. Swap the database for the sample database.
   Refer to the following manual for how to swap the database.
   • YASKAWA Cockpit Add-On Application Software Development Manual (Manual No.: HW1485815)
     Chapter 4.1 YCP Development Task Flow

   A controller is registered in the sample database with the following settings.

<table>
<thead>
<tr>
<th>Database Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factory yaskawa1</td>
</tr>
<tr>
<td>Line a_line</td>
</tr>
<tr>
<td>Cell b_cell</td>
</tr>
<tr>
<td>Controller x</td>
</tr>
</tbody>
</table>

   This controller is set with data items (Item) that have the key name "Torque" and the data type [Double]. Three items of torque data that hold the values {14, 2, 9} are also stored. These items of torque data include timestamps ([long] type) that hold the values 3000000001 to 3000000003.

2. Register the license.
   Refer to the following manual for registering the license.
   • YASKAWA Cockpit INSTRUCTIONS (Manual No.: HW1485838)
     Chapter 4.2 "License Manager"
9.4.2 DA App

The DA App is a C# console application that retrieves a list of torque values from the database and returns the maximum torque value to the database.

9.4.2.1 Execution Sequence Overview

1. Start program
2. Enter user name and password
3. Retrieve token
4. Validate license
   - License: true
5. Set analysis result item (max. torque)
6. Retrieve torque values from database
   - {14, 2, 9}
7. Extract maximum torque
8. Store maximum torque values in database
   - {14}
9. End program
10. Username
11. Password
12. Token
13. GET
14. POST
15. YCP server
9.4.2.2 Execution Sequence

1. Start program
2. Enter user name and password
3. Create new HttpClient
4. Create new HttpRequestMessage
5. Set HttpRequestMessage
   - Method: POST
   - Content:
     ```
     {
       "UserName": "username",
       "Password": "password"
     }
     ```
6. Send HttpRequestMessage
7. Get token
8. Wait for response
9. Get HttpResponseMessage
10. Read response as string
11. Extract token value from string (JSON)
Create new HttpRequestMessage

Set HttpRequestMessage
- Method: GET
- Header:
<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>YCP-API-TOKEN</td>
<td>xxxxxxxx......</td>
</tr>
</tbody>
</table>

Send HttpRequestMessage

Wait for response

Get HttpResponseMessage

{  
  "Result": true,
  "Expiration": "yyyy-mm-dd"
}

License is valid

Validate license
Create new HttpRequestMessage

Set HttpRequestMessage
- Method: POST
- URI:
  http://localhost:8080/YCPApp/api/v2/factories/yaskawa/cells/cell/controllers/x/apps/
  000000000000000000/analysis/item
- Header:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>YCP-API-TOKEN</td>
<td>xxxxxxxx.....</td>
</tr>
</tbody>
</table>

- Content:

```
{
  "Key": "Max_Torque",
  "Type": "Double",
  "Unit": "Nm",
  "Name": "Max_Torque",
  "Visible": false
}
```

Encoding: UTF8
Media type: application/json

Send HttpRequestMessage

Wait for response

Get HttpResponseMessage

YCP server
Create new HttpRequestMessage

Set HttpRequestMessage
- Method : POST
- URI :
  http://localhost:8080/YCPApp/api/v2/factories/yaskawa1
   /lines/a_line/cells/b_cell/controllers/x/data/search
- Header :

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>YCP-API-TOKEN</td>
<td>xxxxxxxxxxxxxx</td>
</tr>
</tbody>
</table>

- Content :

```json
{
  "Item": [
    {"TimeStamp": "2007-03-20T00:00:00Z", "Torque": -48},
    {"TimeStamp": "2007-03-21T00:00:00Z", "Torque": 53},
    {"TimeStamp": "2007-03-22T00:00:00Z", "Torque": -48}
  ],
  "Header": {
    "Returned": 3,
    "QueryResult": 3,
    "ReturnedTotal": 3
  }
}
```

Send HttpRequestMessage

Wait for response

Get HttpResponseMessage

Read response as string

Extract token value from string (JSON)
9.4.3 DW App

The DW App retrieves the torque values and maximum torque values from the database and displays those on the web view. The torque values are also displayed as a graph.
9.4 Sample Program

9.4.3.1 Execution Sequence Overview

1. Open page
2. Validate license
3. Retrieve torque values and maximum torque values from database
4. Display view
5. End script

9.4.3.2 Program Structure

- Auto-refresh every 10 sec
- Open page
- Get token, context path, add-on application name, and locale
  - Token: $[ycpDWToken]
  - Context path: $[contextPath]
  - Add-on application name: $[addOnUIName]
  - Locale: $[locale]
- Load script
- Add scripts to document
- Document ready
9 Appendix

9.4 Sample Program

Document ready

Get factory name, line name, cell name, and controller name

Get language file

Send new AJAX request
- Method : POST
- URI : http://localhost:8080/YCPApp/api/v2/factories/yaskawa1/lines/a_line/cells/b_cell/controllers/x/data/search
- Header :
<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>YCP-DW-TOKEN</td>
<td>$ {ycpDWTOKEN}</td>
</tr>
</tbody>
</table>
- Content :
  ```json
  {
    "Item": ["TimeStamp", "Torque"],
    "N": 10000000,
    "Size": 104857600,
    "Timeout": 2147483647,
    "Query": "Torque IS NOT NULL",
    "Sort": ["Target": "TimeStamp", "Ascending": true]
  }
  ```
- Data type : JSON
- Content type : application/json

Event : success

Get response data (JSON)

```
{
  "Header": {
    "Returned": 3,
    "QueryResult": 3,
    "ReturnedTotal": 3,
  },
  "Data": [
    {
      "TimeStamp": 30000000001, "Torque": 14
    },
    {
      "TimeStamp": 30000000002, "Torque": 12
    },
    {
      "TimeStamp": 30000000003, "Torque": 9
    }
  ]
}
```

Extract torque values
9 Appendix
9.4 Sample Program

Send new AJAX request
- Method: POST
- URI: http://localhost:8080/YCPApp/api/v2/factories/yaskawa1
   /lines/a_line/cells/b_cell/controllers/s/aepp/
   0000000000000000/analysis/search
- Header:
<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>YCP-DW-TOKEN</td>
<td>$ [ycpDWToken]</td>
</tr>
</tbody>
</table>
- Content:
  ```json
  {
    "Item": ["TimeStamp","Max_Torque"],
    "N":1000000,
    "Size":104857800,
    "Timeout":2147483647,
    "Query":"Max_Torque IS NOT NULL",
    "Sort": ["Target":"TimeStamp","Ascending":false]}
  ```
- Data type: JSON
- Content type: application/json

Event: success

Get response data (JSON)
```json
{
  "Header": {
    "Return": 1,
    "QueryResult": 1,
    "ReturnedTotal": 1,
  },
  "Data": [
    {
      "TimeStamp": 123456789, 
      "Torque": 14 
    }
  ]
}
```

Extract max torque value

Display graph

End script
YASKAWA Cockpit
Add-on Application Software
Programmer’s Manual