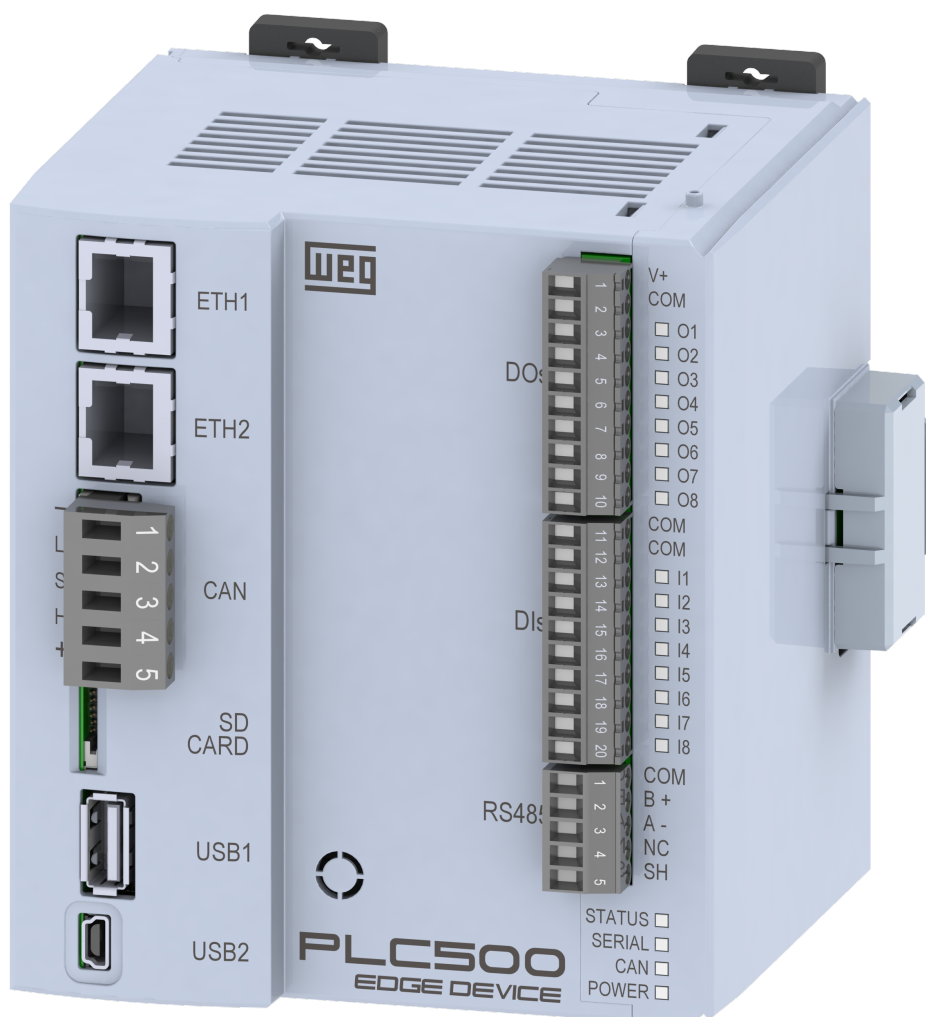


PLC500 EDGE DEVICE

PLC500ED

Nota de Aplicación



Nota de Aplicación

PLC500ED

Documento: 10011077129

Revisión: 00

Fecha de la Publicación: 07/2023

SUMARIO DE LAS REVISIONES

La información abajo describe las revisiones ocurridas en este manual.

Versión	Revisión	Descripción
-	R00	Primera edición.

1	INTRODUCCIÓN	1-1
1.1	ABREVIACIONES Y DEFINICIONES	1-1
1.2	SOBRE EL PLC500ED	1-2
2	CONFIGURACIÓN POR LA WEBPAGE	2-1
2.1	ACCESO AL DISPOSITIVO	2-1
2.1.1	Ethernet o USB	2-1
2.1.2	Requisitos de la Nueva Contraseña	2-2
2.2	PÁGINA DE ESTADO	2-3
2.2.1	Panel de Informaciones del Sistema	2-4
2.2.2	Panel de Conexión MQTT	2-4
2.2.3	Panel de Informaciones del Docker	2-4
2.3	PÁGINA DE CONFIGURACIÓN	2-5
2.3.1	Panel de Integración con la Nube	2-5
2.3.1.1	WEGnology	2-5
2.3.1.2	WEG Smart Machine	2-7
2.3.2	Panel de Interfaces de Red	2-8
2.3.3	Panel de la Interfaz Serial	2-9
2.3.4	Panel Docker	2-10
2.4	PÁGINA DE ADMINISTRACIÓN	2-11
2.4.1	Panel de Acciones de Aplicación	2-11
2.4.2	Panel de Pruebas de Comunicación	2-13
2.4.2.1	Prueba de Comunicación en Serie	2-13
2.4.2.2	Prueba de comunicación TCP/IP	2-14
2.4.2.3	Pruebas de Conectividad del Dispositivo	2-15
2.4.3	Panel de Gestión del Sistema	2-16
3	PLATAFORMA WEGNOLOGY	3-1
3.1	CONFIGURACIÓN INICIAL	3-1
3.1.1	Creación de la Cuenta y de la Aplicación	3-1
3.1.2	Agregar una llave de acceso	3-2
3.1.3	API Token	3-2
3.1.4	Application ID	3-3
3.1.5	Conectando el Dispositivo	3-3
3.1.6	Agregando Atributos	3-3
4	CODESYS - BIBLIOTECA WEGNOLOGY	4-1
4.1	RECOMENDACIONES	4-1
4.2	BIBLIOTECA CODESYS IIOT LIBRARIES SL	4-2
4.3	SINTAXIS DE LOS OBJETOS JSON PUBLICADOS	4-3
4.3.1	WEGnology	4-3
4.3.2	Edge Computing	4-3
4.4	BLOQUES DE FUNCIONES Y MÉTODOS MQTT	4-4
4.5	EJEMPLOS DE APLICACIONES	4-5
4.5.1	Publicación de payloads para la plataforma WEGnology	4-5
4.5.2	Suscripción en tópicos de la plataforma WEGnology	4-8
4.5.3	Publicación de payloads para el Edge Agent	4-11
4.5.4	Suscripción en tópicos del Edge Agent	4-15
5	GUÍA DE INICIO RÁPIDO	5-1
5.1	ESTABLECER CONEXIÓN CON EL PRODUCTO	5-1
5.2	CONECTE EL PRODUCTO A INTERNET A TRAVÉS DE CODESYS O LA PÁGINA WEB	5-1
5.3	CONFIGURAR LA APLICACIÓN IOT A TRAVÉS DE LA PÁGINA WEB	5-2
5.4	HABILITAR UNA IMAGEN DOCKER	5-2
5.5	CREACIÓN DE APLICACIONES	5-3

1 INTRODUCCIÓN

Esta Nota de Aplicación presenta las principales características e informaciones necesarias para la configuración y utilización del PLC500ED.

Es presentada la biblioteca WEGnology, desarrollada por WEG para simplificar el uso de las funcionalidades de conectividad del dispositivo. Para ilustrar algunas posibilidades del producto, son mostrados ejemplos de aplicaciones básicas.

Para más informaciones al respecto del hardware del producto, interfaces y protocolos de comunicación, consulte el Manual del Usuario del PLC500, disponible en www.weg.net.

1.1 ABREVIACIONES Y DEFINICIONES

Broker: Servidor que gestiona la recepción de mensajes enviados por los clientes publisher, enviando a los clientes subscriber, a través del protocolo MQTT.

Codesys: Plataforma de programación que permite desarrollar, configurar y monitorear soluciones para automatización industrial e integración de sistemas.

Container: Instancia de ejecución de una imagen docker conteniendo todos los recursos necesarios para ejecutar una aplicación.

Coreapp: Software de gestión de servicios y recursos del PLC.

DNS: Sistema responsable por la traducción de direcciones IP en nombre de dominios y viceversa (Domain Name System).

Docker: Servicio de software que establece una camada de abstracción para virtualización de sistemas operativos Windows/Linux entregando paquetes llamados containers.

Edge Agent: Container previamente instalado en el producto que permite la ejecución local de Edge Workflows.

Edge Workflow: Lógicas programadas vía plataformas IoT de WEG que son ejecutados localmente en los dispositivos con capacidad de procesamiento de borde.

Ethernet: Arquitectura de interconexión para redes locales (IEEE 802.3).

Gateway: Dispositivo de hardware que permite el flujo de datos entre diversas redes de comunicación.

Imagen Docker: Paquete de software utilizado como template en la generación de containers.

IoT: Sigla que se refiere a las tecnologías que facilitan la comunicación y el intercambio de datos entre dispositivos y la nube, así como entre los propios dispositivos (Internet of Things).

IP: Protocolo utilizado en internet para envío de datagramas entre dispositivos en red (Internet Protocol).

MQTT: Protocolo de transporte que utiliza la topología publicación/suscripción para transferencia de mensajes leves entre dispositivos (Message Queuing Telemetry Transport).

Payload: Contenido del mensaje MQTT.

Plataforma cloud: Plataforma que ofrece un conjunto de servicios de nube, como acceso remoto a software y almacenamiento y procesamiento de datos por medio de internet.

Procesamiento de Borde: Procesamiento de datos realizado cercano al usuario o a la fuente de datos (Edge Computing). Utiliza el container del Edge Agent para la ejecución de los comandos.

QoS: Parámetro utilizado para determinar el nivel de calidad de servicio en intercambio de mensajes, utilizando el protocolo MQTT (Quality of Service).

WEGnology: Plataforma de servicio de nube, utilizada en las aplicaciones de IoT de WEG. Nombre de la biblioteca desarrollada por WEG para uso de las funcionalidades de conectividad del PLC500ED en el CODESYS.

INTRODUCCIÓN

1.2 SOBRE EL PLC500ED

El PLC500 Edge Device (PLC500ED) es un Controlador Lógico Programable con capacidad de Procesamiento de Borde, por el cual es posible conectar equipos industriales a las plataformas de nube de WEG, para utilización en soluciones digitales.

Es desarrollado para atender aplicaciones de mediano y gran porte. Tiene alta velocidad de procesamiento, debido a su CPU compuesta por un procesador Dual-core ARM Cortex-A7 rodando a 1 GHz, un coprocesador Real-time ARM Cortex-M4 de 200 MHz, memoria RAM de 1 GB y Flash de 4 GB.

Tiene un total de 8 salidas digitales, siendo 3 de estas con funcionalidad PWM hasta 300 kHz, y 8 entradas digitales, de las cuales 4 pueden operar hasta 150 kHz.

Como interfaces de comunicación, cuenta con dos puertos Ethernet independientes, puerto CAN, serial RS485, USB OTG, USB device y Micro SD Card

Son utilizados supercondensadores internos para el reloj de tiempo Real (RTC), así como para guardar datos retentivos en la memoria Flash durante el Power Off, dispensando así el uso de baterías.

El PLC500ED permite la conexión de tarjetas de expansión de entradas y salidas digitales, analógicas, termopar, PT100, PT1000, célula de carga, relés, etc., dando más flexibilidad a las aplicaciones. tiene conectores plug-in y su fijación puede ser hecha en riel DIN 35 o directamente en el tablero.

La programación del PLC500ED es realizada por el software CODESYS, ampliamente difundido en el medio industrial, posibilitando la utilización de una infinidad de aplicaciones y funciones ya desarrolladas en el mercado, así como la importación de aplicaciones de otros productos.

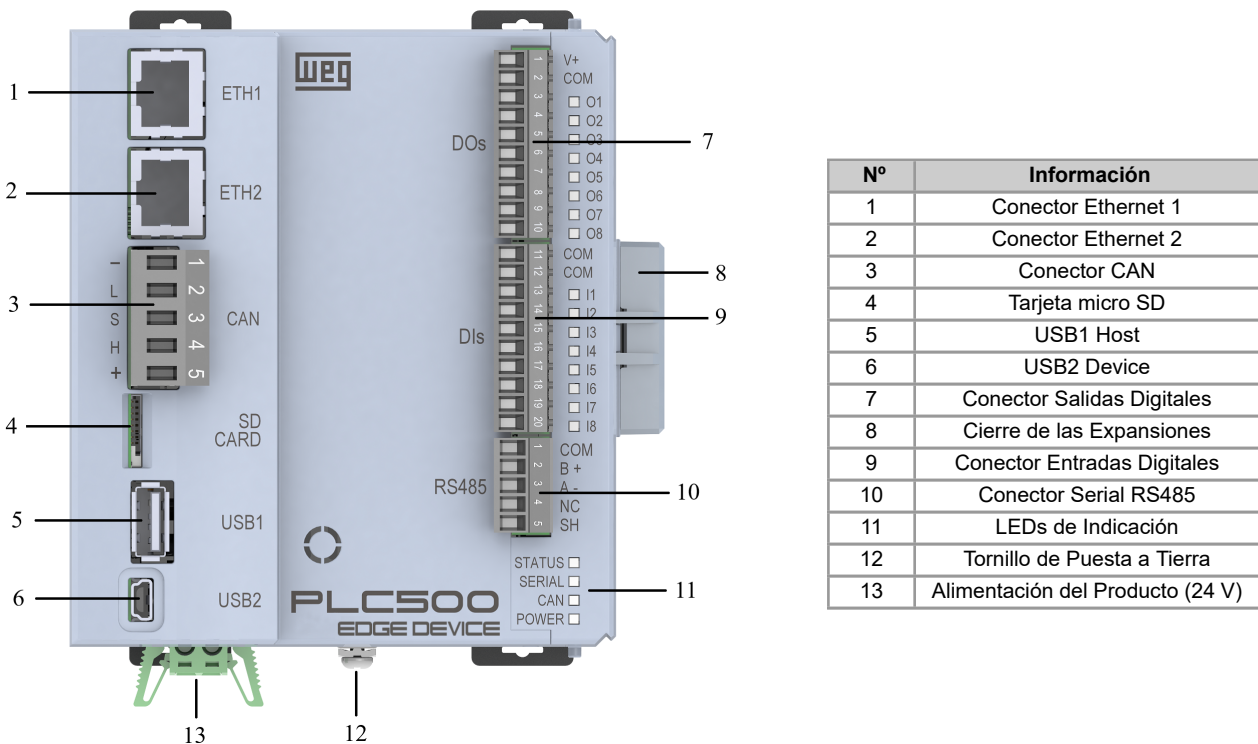


Figura 1.1: PLC500 Edge Device.

2 CONFIGURACIÓN POR LA WEBPAGE

El PLC500ED tiene un servidor web local disponible para comunicación. Es posible acceder a éste a través de las conexiones ETH1, ETH2 y USB2. Para eso, se debe realizar los siguientes pasos:

2.1 ACCESO AL DISPOSITIVO

2.1.1 Ethernet o USB

1. Conectar un cable Ethernet o mini USB entre la computadora y el respectivo puerto del PLC. Los puertos están localizadas en la parte frontal del producto, conforme es visto en la Figura 1.1.
2. La computadora debe ser configurada con un IP estático dentro de la misma red utilizada por la conexión del PLC500ED. Las direcciones de IP estándar de esos puertos de comunicación son mostrados en la tabla 2.1.

Tabla 2.1: Direcciones de IP estándar PLC500ED.

Conexión	Dirección de IP estándar
ETH1	192.168.1.10
ETH2	192.168.2.10
USB2	192.168.234.234

Ejemplo de configuración (considerando los IPs estándar): configure la dirección de IP de la Computadora como 192.168.1.X (ETH1), 192.168.2.X (ETH2) o 192.168.234.Y (USB2), donde X es cualquier valor entero de 1 a 254, excepto 10, donde Y es cualquier valor entero de 1 a 254, excepto 234. Deje la máscara de red en su valor estándar (255.255.255.0).



¡NOTA!

Las direcciones de IP de los puertos ETH1 y ETH2 pueden ser modificadas a través del **Codesys** o en el menú **Configuration** de la Webpage del producto.



¡NOTA!

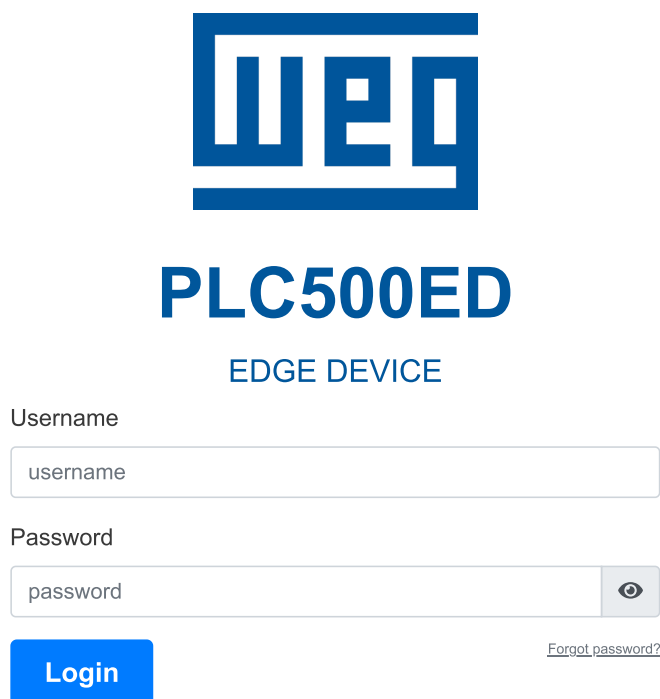
El IP del USB2 es fijo y no puede ser modificado.



¡ATENCIÓN!

Al bajar una aplicación por el **Codesys**, la configuración del **Setup** es grabada en el producto, sobrescribiendo cualquier configuración realizada por la **Webpage**.

3. Una vez que la configuración esté completa, será necesario utilizar un navegador web reciente (preferentemente Firefox o Chrome) y digitar el IP del puerto del dispositivo en el campo de URL. Luego de realizar la conexión vía Ethernet o USB, deberá aparecer en el navegador una página con la pantalla de login, conforme lo muestra la Figura 2.1.



Username

Password

[Forgot password?](#)

Login

Figura 2.1: Pantalla de login del PLC500ED.

4. Inicialmente, el dispositivo viene configurado con el login y la contraseña estándar de WEG, siendo necesaria su actualización en el primer acceso. El login y la contraseña estándar son:

- Usuario: weg
- Contraseña: weg

2.1.2 Requisitos de la Nueva Contraseña

En el primer intento de acceso será exigida la modificación de usuario y la contraseña. La nueva contraseña debe respetar los siguientes criterios de seguridad:

- Tener como mínimo 14 caracteres
- Tener como mínimo 1 carácter mayúsculo
- Tener como mínimo 1 carácter numérico
- Tener como mínimo 1 carácter especial (símbolo)

Luego de rellenar los campos de la ventana de generación de nuevo login y hacer clic en el botón **Submit**, aparecerá un mensaje de éxito, si los criterios descritos anteriormente fueron respetados. Luego de este procedimiento será posible hacer el login normalmente, digitando las nuevas credenciales.



¡ATENCIÓN!

Se recomienda **guardar** el nuevo **login** y **contraseña** en local seguro, ya que la recuperación de login y/o contraseña solamente será posible vía **soporte técnico!**

2.2 PÁGINA DE ESTADO

Una vez logado, el usuario tendrá acceso a la Página de Estado del producto. Dentro de esta página es posible visualizar informaciones relativas al funcionamiento del sistema, conexión MQTT y aplicaciones Docker, a través de tres paneles, siendo ellos:

- Panel de Informaciones de Sistema
- Panel de Conexión MQTT
- Panel de Informaciones del Docker

La Figura 2.2 muestra la Página de Estado del PLC500ED.

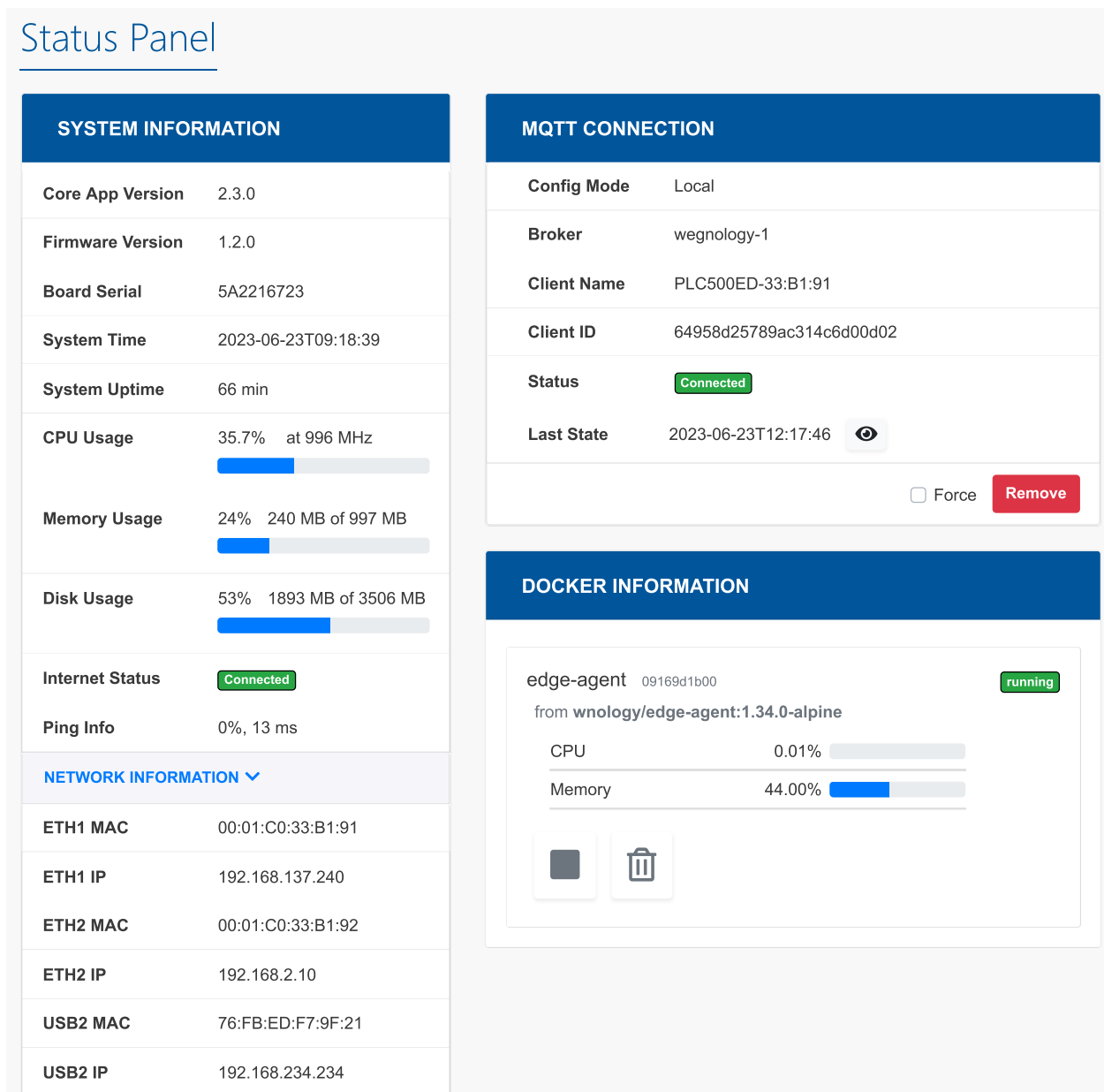


Figura 2.2: Página de Estado.

CONFIGURACIÓN POR LA WEBPAGE

2.2.1 Panel de Informaciones del Sistema

En el Panel de Informaciones del Sistema es posible visualizar informaciones sobre versiones, modelo, configuraciones, estados, utilización de recursos, entre otros. La lista a continuación presenta la descripción detallada de cada campo contenido en el panel.

- **Core app Version:** Versión del software relacionado al Core app en el PLC500ED.
- **Codesys Version:** Versión del software relacionado al Codesys en el PLC500ED.
- **Board Serial:** Identificador único asignado al circuito electrónico.
- **System Time:** Fecha y hora del sistema.
- **System Uptime:** Tiempo de operación del sistema, desde su inicialización (expresado en minutos).
- **CPU Usage:** Utilización de la CPU y frecuencia actual del procesador (expresadas en % y MHz).
- **Memory Usage:** Cantidad de memoria RAM siendo utilizada en el momento (expresada en % y MB).
- **Disk Usage:** Cantidad de espacio en disco ocupado en el momento (expresado en % y MB).
- **Internet Status:** Estado de la conexión a internet (Conectado o Desconectado).
- **Ping Info:** Tasa de pérdida de paquetes registrados (expresada en %) y Round-trip time - Tiempo entre la solicitud y la recepción de un paquete registrado (expresado en milisegundos).

Informaciones de red:

- **ETH1 MAC:** Dirección MAC de la interfaz Ethernet ETH1.
- **ETH1 IP:** Dirección IP de la interfaz Ethernet ETH1.
- **ETH2 MAC:** Dirección MAC de la interfaz Ethernet ETH2.
- **ETH2 IP:** Dirección IP de la interfaz Ethernet ETH2.
- **USB2 MAC:** Dirección MAC de la interfaz USB.
- **USB2 IP:** Dirección IP de la interfaz USB.

2.2.2 Panel de Conexión MQTT

Una vez que el PLC500ED esté conectado a la plataforma WEGnology, será posible visualizar el panel de conexión MQTT, con las siguientes informaciones:

- **Config Mode:** Modo de configuración: Local o Remoto.
- **Broker:** Nombre del broker MQTT.
- **Client Name:** Nombre del cliente MQTT.
- **Client ID:** Identificador del cliente MQTT.
- **Status:** Estado de la conexión MQTT (Conectado o Desconectado).
- **Last State:** Información de timestamp del último payload de estado enviado.
- **Force (caja de selección):** Flag que fuerza la remoción de la conexión MQTT.
- **Remove (botón)** tiene la función de remover la conexión MQTT entre el cliente y el broker. Esta funcionalidad está disponible solamente cuando el integrador WEGnology es seleccionado en la Página de Configuración.



¡NOTA!

Para visualizar las informaciones del **Panel de Conexión MQTT** es necesaria la configuración de la aplicación en el **Panel de Integración con la Nube**, ubicado en la **Página de Configuración**.

2.2.3 Panel de Informaciones del Docker

En el Panel de Informaciones del Docker son presentadas las siguientes informaciones:

- **Identificación del contenedor:** Nombre del container, su respectivo short ID y la imagen base usada para el montaje del container, localizados en el ángulo superior izquierdo.
- **Estado del container:** Estado actual del container, ubicado en el ángulo superior derecho (inicializando, rodando o parado).
- **CPU y Memory:** Porcentaje de la utilización de CPU y de memoria RAM relacionadas al container.
- Los botones identificados por un cuadrado y un cubo de residuos tienen las funciones de parar y remover el container, respectivamente.



¡NOTA!

Para visualizar las informaciones del **Panel de Informaciones del Docker** es necesario habilitar la imagen del container específico no **Panel del Docker**, ubicado en la **Página de Configuración**.

2.3 PÁGINA DE CONFIGURACIÓN

La Página de configuración permite la configuración de parámetros relacionados a la red, plataforma de nube, interfaz serial e imágenes Docker. Dentro de esta página es posible acceder a los siguientes paneles:

- Panel de Integración con la Nube
- Panel de Interfaces de Red
- Panel de Interfaz Serial
- Panel Docker

La Figura 2.3 muestra la Página de Configuración del PLC500ED.

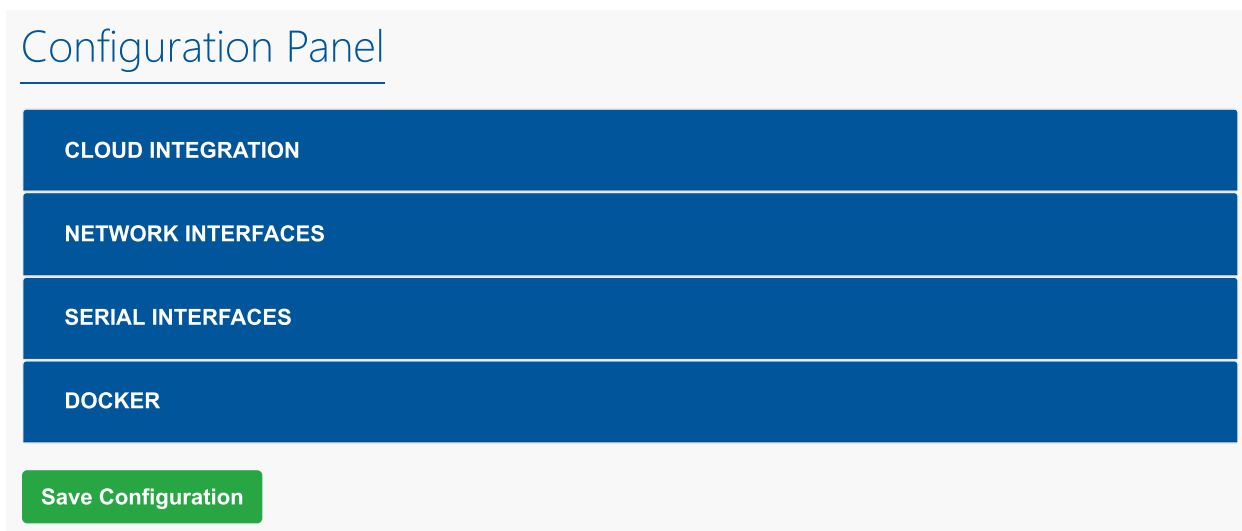


Figura 2.3: Página de Configuración.

2.3.1 Panel de Integración con la Nube

En el Panel de Integración con la Nube es posible seleccionar dos integradores diferentes: **WEGnology** y **WEG Smart Machine**. Los demás parámetros del panel dependen del integrador seleccionado.

2.3.1.1 WEGnology

La Figura 2.4 muestra los parámetros de configuración disponibles para el integrador WEGnology.

Configuration Panel

CLOUD INTEGRATION

Integrator: MQTT Configuration Status: **Locally Configured**

Application ID: API Token: Token file: (optional) Nenhum arq... escolhido

Access Key: Secret: Access Keys file: (optional) Nenhum arq... escolhido

NETWORK INTERFACES

SERIAL INTERFACES

DOCKER

Figura 2.4: Página de Configuración - Integración con la Nube - WEGnology.



¡NOTA!

- Antes de configurar la conexión con alguna plataforma cloud de WEG, en la webpage del PLC500ED, es necesario realizar los procedimientos de generación de la aplicación y de credenciales en la plataforma respectiva.
- Para más informaciones sobre estos procedimientos, consultar la documentación referente a la plataforma en el sitio de WEG o consultar al soporte técnico.

- **Integrator:** Servicio de integración de nube (Seleccionar opción: **WEGnology**).
- **Application ID:** ID de la aplicación existente en la plataforma de la nube.
- **API Token:** Token para permiso de gestión de la aplicación en la plataforma de la nube.
- **Token file:** Opción de importar archivo conteniendo las informaciones de API Token. En caso de que esta opción sea usada, el campo anterior no precisará ser rellenado.
- **Access Key:** Llave de acceso para permitir que un dispositivo se conecte a la plataforma de la nube.
- **Secret:** Contraseña de acceso para permitir que un dispositivo se conecte a la plataforma de la nube.
- **Access Keys file:** Opción de importar archivo conteniendo las informaciones de Access Key y Secret. En caso de que esta opción sea usada, los dos campos anteriores no precisarán ser rellenados.

2.3.1.2 WEG Smart Machine

La Figura 2.5 muestra los parámetros de configuración disponibles para el integrador WEG Smart Machine.

Configuration Panel

CLOUD INTEGRATION

Integrator: WEG Smart Machine v1

MQTT Configuration Status: **Locally Configured**

Application ID: Application ID

API Token: Token

Device ID: Device ID

Access Key: Keys

Secret: Secret

Credentials file: (optional) Escolher arquivo Nenhum arquivo escolhido

NETWORK INTERFACES

SERIAL INTERFACES

DOCKER

Save Configuration

Figura 2.5: Página de Configuración - Integración con la Nube - WEG Smart Machine.

- **Integrator:** Servicio de integración de nube (Seleccionar opción: **WEG Smart Machine**).
- **Application ID:** ID de la aplicación existente en la plataforma de la nube.
- **API Token:** Token para permiso de gestión de la aplicación en la plataforma de la nube.
- **Device ID:** ID del dispositivo registrado en la plataforma de la nube.
- **Access Key:** Llave de acceso para permitir que un dispositivo se conecte a la plataforma de la nube.
- **Secret:** Contraseña de acceso para permitir que un dispositivo se conecte a la plataforma de la nube.
- **Credentials file:** Opción de importar archivo conteniendo todas Las informaciones descritas arriba. En caso de que esta opción sea usada, los cinco campos anteriores no precisarán ser rellenados.

CONFIGURACIÓN POR LA WEBPAGE

2.3.2 Panel de Interfaces de Red

En el Panel de Interfaces de Red es posible configurar las interfaces ETH1 y ETH2, como puede ser visto en la Figura 2.6. Cada interfaz puede ser habilitada o deshabilitada a través de los botones On/Off en el ángulo derecho del panel, alineados con el nombre de cada interfaz.

The screenshot shows a web-based configuration panel titled "Configuration Panel". It has a dark blue header with "CLOUD INTEGRATION" and "NETWORK INTERFACES" sections. Below "NETWORK INTERFACES", there are three interface configuration sections: "Interface ETH1", "Interface ETH2", and "Interface USB2". Each section has a status toggle (On/Off) in the top right corner. The "Interface ETH1" section has "Use DHCP" and "Default Route" dropdowns set to "Yes", and input fields for "IP Address" (placeholder "IP"), "Network Mask" (placeholder "Mask"), and "Gateway" (placeholder "Gateway"). The "Interface ETH2" section has "Use DHCP" and "Default Route" dropdowns set to "No", and input fields for "IP Address" (192.168.2.10), "Network Mask" (255.255.255.0), and "Gateway" (placeholder "Gateway"). The "Interface USB2" section has input fields for "IP Address" (192.168.234.234) and "Network Mask" (255.255.255.0). At the bottom, there are sections for "SERIAL INTERFACES" and "DOCKER", and a green "Save Configuration" button.

Figura 2.6: Página de Configuración - Interfaces de Red.

La lista a continuación presenta la descripción de cada campo a ser configurado por el usuario.

Interfaces ETH1 y ETH2

- **Use DHCP:** Habilita/Deshabilita la utilización de DHCP.
- **Default Route:** Habilita/Deshabilita la ruta estándar de red para la dirección de destino de los paquetes IP.
- **IP Address:** Dirección IP de la interfaz Ethernet.
- **Network Mask:** Máscara de red referente a la dirección IP de la interfaz Ethernet.
- **Gateway:** Dirección IP del Gateway de la red.



¡NOTA!

No configure las interfaces ETH1 y ETH2 con IPs de una misma red. Ej: 192.168.1.10 y 192.168.1.20. En caso de que lo haya hecho, se podrá entrar en la página web, vía USB2, y cambiar los IPs nuevamente con los valores correctos, o realizar el **Factory Reset**.



¡NOTA!

En caso de que la configuración del **acceso a internet** sea realizada por medio de una interfaz Ethernet, recuerde activar la opción **Default Route**.

Interfaz USB2

En el panel de la interfaz USB2 son mostrados la respectiva dirección de IP y la máscara de red. Sin embargo, estos datos constan en la Página de Configuración, solamente a título de información, una vez que sus campos no pueden ser alterados.

2.3.3 Panel de la Interfaz Serial

En este panel es posible habilitar o deshabilitar la interfaz serial vía RS485, para ser utilizada a través de los containers Docker. La Figura 2.7 muestra el Panel de la Interfaz Serial.

Configuration Panel

The screenshot shows a configuration panel with several sections: CLOUD INTEGRATION, NETWORK INTERFACES, SERIAL INTERFACES, and DOCKER. Under SERIAL INTERFACES, there is a section for 'Interface RS485' with a toggle switch set to 'Off'. At the bottom of the panel, there is a green 'Save Configuration' button.

Figura 2.7: Página de Configuración - Panel de la Interfaz Serial.



¡ATENCIÓN!

No es recomendada la utilización concomitante de la RS485 vía plataformas WEG o por el Codesys, una vez que podrán ser generados errores y la comunicación podrá ser interrumpida.

CONFIGURACIÓN POR LA WEBPAGE

2.3.4 Panel Docker

En el Panel Docker, visto en la Figura 2.8, es posible visualizar las siguientes informaciones:

Configuration Panel

CLOUD INTEGRATION

NETWORK INTERFACES

SERIAL INTERFACES

DOCKER

Image Name	ID	Size	Created at	Status
wnology/edge-agent:1.25.0-alpine	4e41711b3	525	2021-11-24T13:56:35	Enabled <input type="button" value="Off"/>

Save Configuration

Figura 2.8: Página de Configuración - Panel Docker.

- **Image Name:** Nombre de las imágenes Docker usadas para la construcción de los containers.
- **ID:** Short ID de las imágenes Docker.
- **Size:** Tamaño de las imágenes Docker en MB.
- **Created at:** Fecha y hora de la creación de las imágenes Docker.
- **Status:** Estado de la imagen Docker - Identifica si hay algún container creado a partir de la imagen. Es posible habilitar o deshabilitar containers a partir de las imágenes, a través de los botones On/Off.



¡NOTA!

Para que el **Panel de Informaciones del Docker** sea visualizado en la **Página de Estado**, se debe habilitar la imagen Docker en el campo **Status**.

2.4 PÁGINA DE ADMINISTRACIÓN

La Página de Administración también puede ser accedida por el usuario luego del login. Dentro de esta página es posible visualizar los siguientes paneles:

- Panel de Acciones de Aplicación
- Panel de Pruebas de Comunicación
- Panel de Gestión del Sistema

La Figura 2.9 muestra la Página de Administración del PLC500ED.

The screenshot shows the 'Administration Panel' interface. It is divided into three main sections, each with a blue header bar:

- APPLICATION ACTIONS:** Contains five items: 'Restart App' (with a refresh icon), 'Factory Reset' (with a factory icon), 'Change Login' (with a login icon), 'Software Update' (with a refresh icon), and 'System Log Files' (with a document icon).
- COMMUNICATION TESTS:** Contains three items: 'Serial Communication Test', 'TCP/IP Network Communication Test', and 'Check Device Networks Connectivity'. Each item has a green 'Start' button.
- SYSTEM MANAGEMENT:** Contains two items: 'VPN Client Service' (with a 'Disconnected' status indicator and a VPN icon) and 'SSH Server' (with an 'Off' toggle switch).

Figura 2.9: Página de Administración.

2.4.1 Panel de Acciones de Aplicación

En el Panel de Acciones de Aplicación existen cinco botones que permiten al usuario: reinicializar la aplicación, restaurar las configuraciones de fábrica, modificar el login de la página web, actualizar el firmware del producto y bajar los logs del sistema. Abajo pueden ser vistos más detalles.

- **Restart App:** Esta opción solamente reinicia las aplicaciones Edge del producto, sin reiniciar el PLC. Luego, la aplicación del Codesys no será reiniciada.



¡NOTA!

A pesar de que la aplicación del Codesys no sea reiniciada, la utilización del **Restart App** puede impactar en la aplicación en ejecución, una vez que los servicios ligados a las plataformas WEG serán reiniciados.

CONFIGURACIÓN POR LA WEBPAGE

- **Factory Reset:** Al hacer clic en este botón, serán tomadas las siguientes acciones, seguidas de la reinicialización del producto:
 - Modificación de login y contraseña de la página web, al estándar de fábrica (login: weg, contraseña: weg).
 - Retomada de todas las configuraciones realizadas en la Página de Configuración al estándar de fábrica (cloud, redes, Docker).
 - Restauración de las configuraciones de IP de las redes a los valores estándar.
 - Exclusión de la aplicación del Codesys.
- **Change Login:** Esta opción resetea el usuario y la contraseña al estándar de fábrica (login: weg, contraseña: weg).
- **Software Update:** Al presionar el botón **Software Update**, se abrirá una ventana mostrando las opciones de **Actualización de Firmware** del producto, con el **Update Mode** inicialmente en el modo **Local**, conforme lo muestra la Figura 2.10(a). No obstante, la opción **Local** puede ser alterada a **Remote**, de acuerdo con la Figura 2.10(b).

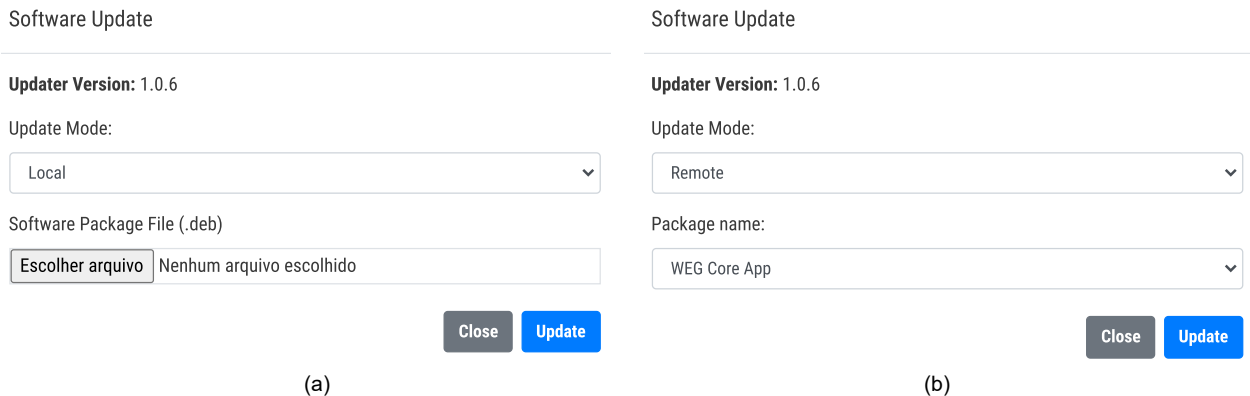




Figura 2.10: Panel de Actualización de Firmware: (a) Actualización Local. (b) Actualización Remota.

- **Modo Local:** La actualización de firmware en modo local permite que las actualizaciones puedan ser realizadas aunque el PLC500ED no esté conectado a internet. Para que eso sea posible, es necesario que el paquete de firmware sea previamente bajado en la computadora o en el dispositivo móvil por donde la página web está siendo accedida. En seguida, bastará hacer el upload del paquete, por medio del botón **Choose File** y, a continuación, presionar el botón **Update**.

 **¡NOTA!** En la actualización de firmware en **Modo Local**, pueden ser escogidos tanto el archivo **WEG Core App** como el **PLC500ED Firmware**.

- **Modo Remoto:** La actualización de firmware en modo remoto permite que la actualización pueda ser realizada sin que el paquete de firmware sea previamente bajado. En la opción **Package name** pueden ser escogidos los paquetes **WEG Core App** (cuestiones relacionadas a la gestión de servicios) y **PLC500ED Firmware** (cuestiones relacionadas básicamente al Codesys).

Con estas condiciones cumplidas, basta presionar el botón **Update** para iniciar la actualización. Para eso, es esencial que el PLC500ED esté con **acceso a internet**.

 **¡NOTA!** En la opción **PLC500ED Firmware**, el producto será reiniciado. En la opción **WEG Core App**, solamente a las aplicaciones ligadas a las plataformas WEG serán reiniciadas.

- **System Log Files:** Para que sea posible bajar los logs del sistema, es exigido el ingreso de una llave de autenticación, antes de hacer el download. En caso de que esta opción sea necesaria, entrar en contacto con WEG para obtener esta llave de autenticación.

2.4.2 Panel de Pruebas de Comunicación

En el Panel de Pruebas de Comunicación es posible realizar las siguientes pruebas:

- Prueba de Comunicación en Serie
- Prueba de Comunicación TCP/IP
- Pruebas de Conectividad del Dispositivo

2.4.2.1 Prueba de Comunicación en Serie

En el Panel de Pruebas de Comunicación es posible realizar la prueba de la comunicación **Modbus-RTU**. Al presionar el botón **Start**, se abrirá la ventana presentada en la Figura 2.11. Como puede ser visto, es posible configurar los parámetros necesarios para establecer la comunicación Modbus-RTU vía RS485 y realizar la operación de **Read Holding Registers**. La lista a seguir presenta la descripción de cada campo a ser configurado por el usuario.

Test Protocol: Modbus RTU

Communication Parameters

Unit ID:

Speed:

Bits:

Parity:

Stop bits:

Holding Registers Read Information


Start Address: Size: 

Figura 2.11: Panel de Pruebas de Comunicación.

- **Unit ID:** Número identificador del dispositivo.
- **Speed:** Tasa de transferencia o baud rate de la comunicación serial.
- **Bits:** Bits de datos.
- **Parity:** Bits de paridad.
- **Stop Bits:** Bits de parada.
- **Start Address:** Dirección del primer registrador a ser leído.
- **Size:** Número de registradores a ser leídos a partir del dirección inicial.

El botón **Add Registers** permite que un nuevo conjunto de campos **Start Address** y **Size** pueda ser rellenado para incluir más intervalos de registradores a ser leídos. El botón con símbolo de cubo de residuos borra los respectivos campos Start Address y Size creados. Luego de rellenar todos los campos, bastará presionar el botón **Execute** para realizar la prueba.



¡NOTA!

No es recomendado utilizar la **Prueba de Comunicación en Serie** cuando sea utilizada la **red RS485** vía Codesys, una vez que la comunicación podrá ser interrumpida.

CONFIGURACIÓN POR LA WEBPAGE

2.4.2.2 Prueba de comunicación TCP/IP

Esta función permite realizar pruebas de conectividad de red TCP/IP entre el producto y otros destinos. Al presionar el botón **Start**, se abre la ventana Figura 2.12.



¡NOTA!

La **Prueba de comunicación TCP/IP** prueba la conectividad de las redes TCP/IP. No es posible leer o escribir direcciones de memoria a través del protocolo Modbus.

Communication Parameters

Host/IP Address:

Port:

Timeout:

Network Communication Tests

Ping Test



Open Port Test



Close

Figura 2.12: Prueba de Comunicación TCP/IP.

Hay dos tipos de pruebas posibles, que se pueden utilizar para comprobar las redes de área local (LAN) y las redes conectadas mediante Internet (WAN):

- **Ping Test:** la prueba de ping utiliza el protocolo ICMP para verificar que el producto puede llegar al destino especificado en el campo Host/IP Address. En este campo se permite especificar el dominio o IP de destino. Si se puede llegar al destino, aparecerá un símbolo de verificación verde junto al botón de prueba. De lo contrario, un símbolo "X" rojo indica que no se puede alcanzar el destino.



¡NOTA!

Es importante recordar que, en algunos casos, aunque el destino sea accesible en la red, el protocolo ICMP puede ser bloqueado por un firewall. Por lo tanto, también es importante realizar la prueba **Open Port Test** para garantizar el resultado.

- **Open Port Test:** la prueba de puerto abierto realiza una conexión de socket entre el producto y el destino especificado por los campos Host/IP Address + Port. Si se establece el socket, aparecerá un símbolo de verificación verde junto al botón de prueba. De lo contrario, un símbolo "X" rojo indicará que no se puede establecer el socket. En este segundo caso, si la prueba de ping fue exitosa, es probable que el puerto del lado de destino no esté disponible para la conexión.

Los campos que debe configurar el usuario son los siguientes:

- **Host/IP Address:** Dirección IP de destino.
- **Port:** Puerto de comunicación.
- **Timeout:** Tiempo de espera en segundos. Es configurable para ambas pruebas.

2.4.2.3 Pruebas de Conectividad del Dispositivo

Esta funcionalidad permite verificar si los principales destinos de Internet, necesarios para el correcto funcionamiento del PLC500ED, son accesibles con las configuraciones de red actuales.

Todos los servicios especificados debajo del campo **Mandatory remote services** deben tener un símbolo de verificación verde, que indica que el servicio es accesible. Si algún campo muestra un símbolo "X" rojo, se debe tomar alguna acción para corregir el problema. La figura 2.13 muestra la ventana que se abre al presionar el botón **Start**, donde se puede ver que todos los campos son accesibles.

Mandatory remote services

DNS name resolution ✓

WEGnology Web Server ✓

WEGnology MQTT Broker ✓

WEGnology REST API Server ✓

Docker Container Repository ✓

WEG Software Update Repository ✓

Close

Check

Figura 2.13: Pruebas de Conectividad del Dispositivo.

Cuando algún destino no pasa la prueba, puede ser que haya algún cortafuegos en la red que bloquee el acceso al dominio especificado. Para más detalles sobre los destinos y dominios que deben ser accesibles y liberados para que el producto pueda usarse en Internet, consulte la Tabla 2.3.

Tabla 2.3: Destinos y dominios necesarios para la liberación del firewall.

Service name	Function	Description	Protocol	Port	Destination	Destination IP
WEGnology MQTT Broker	Communication with MQTT Broker	Used to transmit measurements and commands from the Edge device to the MFM platform	TCP / MQTT	8883	broker.app.wnology.io	3227206235 3.234.136.81 52.22.246.163
WEGnology REST API Server	WEGnology applications	Services of the application in which the device is connected and also of the WEGnology platform	TCP / HTTPS	443	*.wnology.io api.app.wnology.io	3.227.206.235 3.234.136.81 52.22.246.163
WEGnology Web Server	Internet connection status	Used for checking internet status and plant configuration via WEG Scan webpage	TCP / ICMP	443	console.app.wnology.io	65.8.248.99 65.8.248.109 65.8.248.31 65.8.248.46
Docker Container Repository	Communication with the Drive Scan application repository	Drive Scan application repository	TCP / HTTPS	443	hub.docker.com	Dynamic IP address
WEG Software Update Repository	Communication for Drive Scan Firmware Update	Drive Scan Firmware Update	TCP / HTTPS	443	nexus3.weg.net	Dynamic IP address
DNS name resolution	Lookup of IP addresses by DNS	Required when the customer system could not resolve the domain names	TCP / UDP	53	-	8.8.8.8 8.8.4.4
-	Internet connection status	Used for checking internet status and plant configuration via WEG Scan webpage	ICMP	53	google.com	Dynamic IP address

2.4.3 Panel de Gestión del Sistema

En el Panel de Administración del Sistema son encontradas las funciones **VPN Client Service** y **SSH Server**, que deben ser utilizadas solamente por el soporte técnico de WEG.

3 PLATAFORMA WEGNOLOGY

Este capítulo presenta los pasos básicos para la configuración del usuario y de la aplicación en la plataforma WEGnology. Para mayores detalles, favor consultar la documentación específica en docs.app.wnology.io.

Conforme es mostrado en la Figura 2.4 del capítulo anterior, para la integración con la nube, son necesarios los siguientes datos, a ser ingresados en la página web del producto:

- Access Key
- Secret
- API Token
- Application ID

Estos datos pueden ser ingresados manualmente, o a través del upload de archivos específicos que contengan esas informaciones. En la próxima sección son mostrados los pasos básicos para la configuración inicial de la plataforma y la obtención de éstos.

3.1 CONFIGURACIÓN INICIAL

3.1.1 Creación de la Cuenta y de la Aplicación

Primeramente, se debe entrar en la página accounts.app.wnology.io y crear una cuenta, ingresando los datos necesarios. A continuación, es que se dé un nombre a la aplicación y, opcionalmente, ingresar una descripción. La Figura 3.1 muestra las pantallas de creación de cuenta (Figura 3.1(a)) y creación de la aplicación (Figura 3.1(b)).

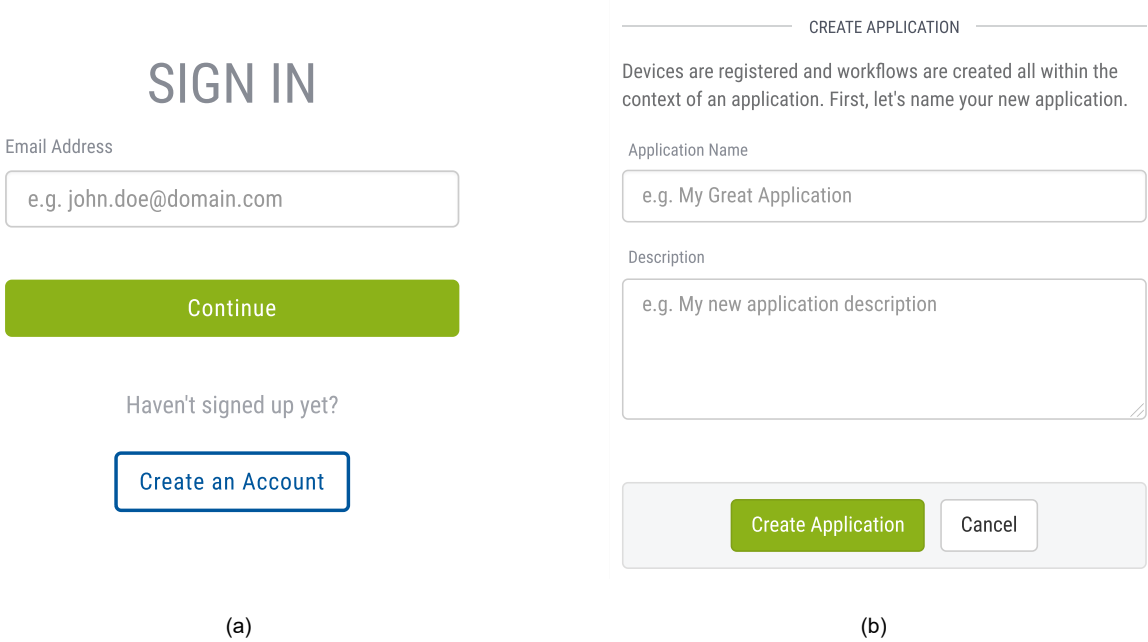


Figura 3.1: Creación de la cuenta y de la aplicación en la plataforma WEGnology. (a) Página de login. (b) Página de creación de la aplicación.

Luego de crear la aplicación, es abierta la pantalla principal de la aplicación WEGnology, mostrada en la Figura 3.2.

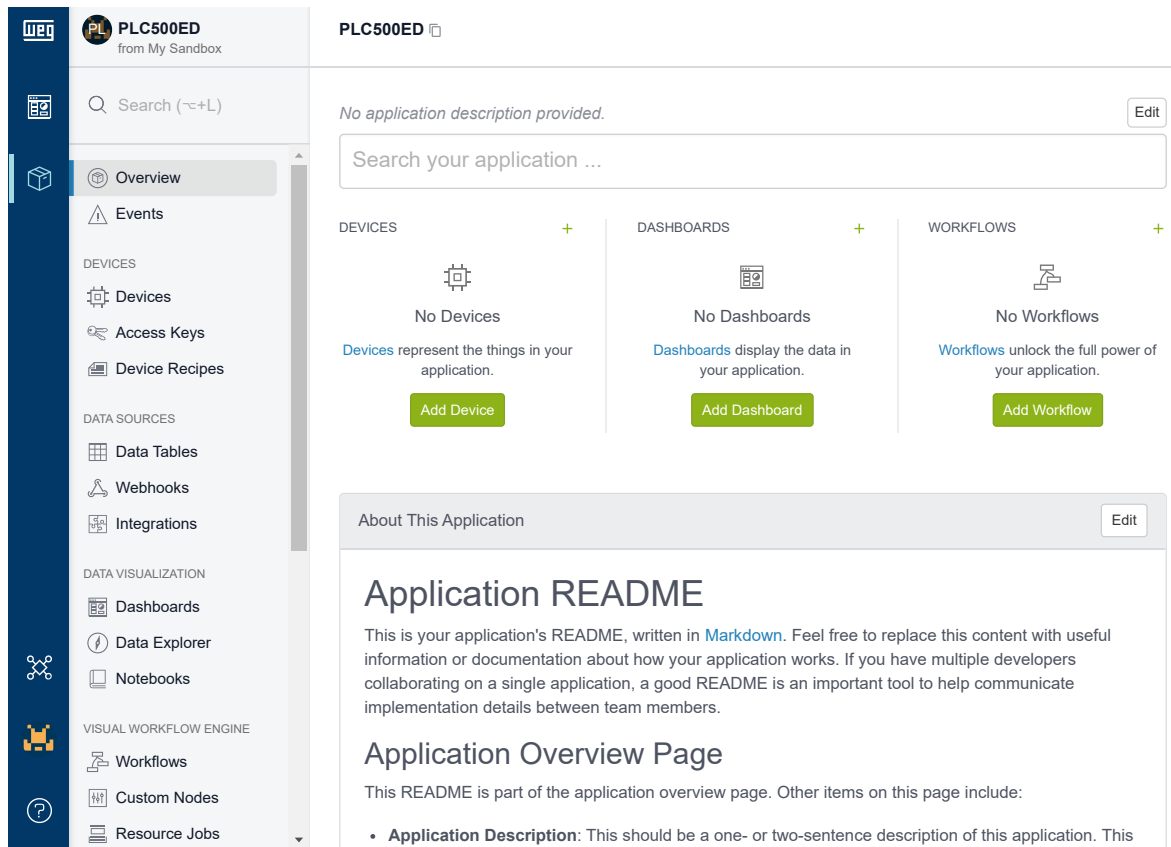


Figura 3.2: Pantalla principal de la aplicación WEGnology.

3.1.2 Agregar una llave de acceso

En la pantalla principal, seleccionar **Access Keys > Add Access Key > Create Access Key**. serán creados la **Access Key** y el **Access Secret**, que deben ser anotados o guardados a través del archivo generado por el **Download File**. La Figura 3.3 muestra un ejemplo de llave y de secreto creados.

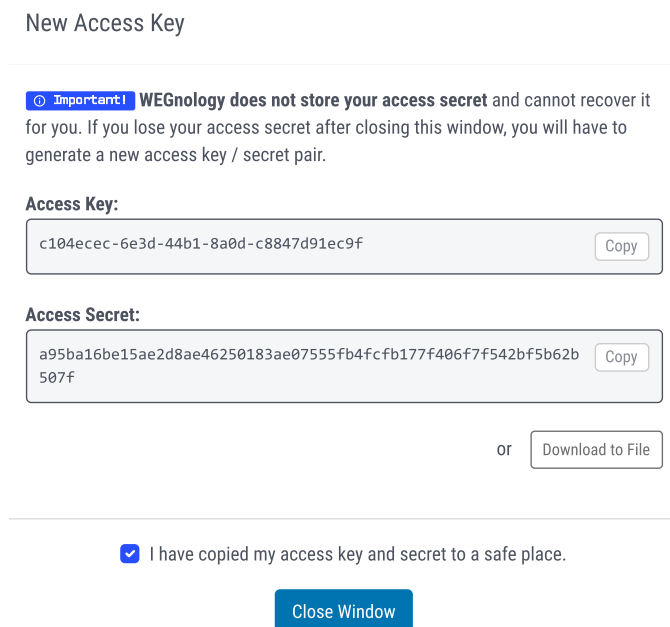


Figura 3.3: Creando una nueva llave de acceso.

3.1.3 API Token

Para la obtención del **Token**, seleccionar **API Tokens > Add Application Token > Create Application Token**. Será creado un nuevo **Application API Token**, que debe también ser anotado o guardado a través del archivo generado por el **Download File**. La Figura 3.4 muestra un ejemplo de Application API Token creado.

New Application Token

Important! WEGnology does not store your application token and cannot recover it for you. If you lose your application token after closing this window, you will have to generate a new application token.

Application API Token:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2M2M1OWVhMmJhMGM3NjI3YTU3Yjc5YzYzEiLCJzZW50eXB1IjoieXBpVG9rZW4iLCJzY29wZSI6WyJhbGwvQXBwbG1jYXRpb24iXSwiaWF0IjoxNjc3ODk1NTg2LCJpc3MiOiJhcHAud25vbG9neS5pbyJ9.OSBvtkb61QAEbVehv8_g2XmwAbFHRCJ4KjQfBP8kyQ
```

or

I have copied my application token to a safe place.

Figura 3.4: Creando un nuevo API Token.

3.1.4 Application ID

Se puede encontrar el **Application ID** en **Overview** y luego hacer clic en el icono de copia junto al nombre de la aplicación. También aparece al final de la URL de la página, como se muestra en la figura 3.5.

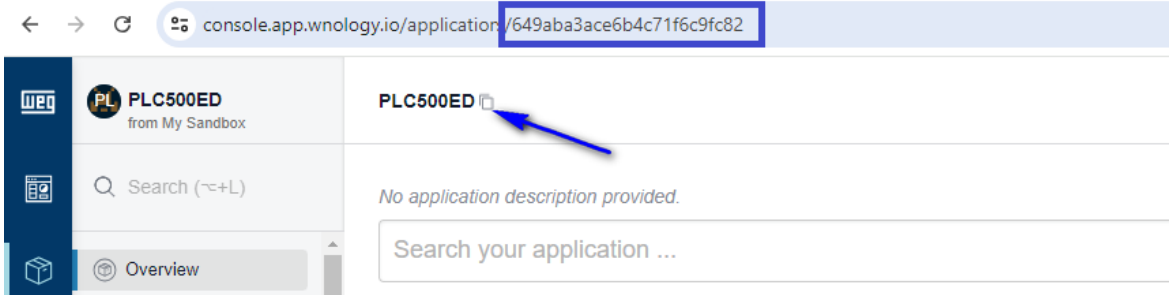


Figura 3.5: Obteniendo el Application ID.

3.1.5 Conectando el Dispositivo

Con los datos obtenidos encima en manos, ahora se puede acceder a la página web del producto y ingresarlos en el **Panel de Integración con la Nube**, en la Página de Configuración, y guardar la configuración. Si son ingresados los datos correctos, la Página de Estado mostrará el **Panel de Conexión MQTT** con los datos de la aplicación WEGnology.

Luego de estos pasos, el dispositivo debe ser conectado automáticamente a la plataforma WEGnology. Para hacer la verificación, se debe hacer clic en **Devices** y verificar si el dispositivo fue agregado correctamente. El ícono de un plug verde indica que el dispositivo está conectado. La Figura 3.6 muestra el dispositivo conectado a la plataforma WEGnology.

Connection Status	Name	Device Class	Last Updated
Any Status	Filter by Name...	Any Class	Select...
Connected since 17 de jan de 2023 10:08:36	PLC500ED-21:B0:1F ID: 63c69dabec4a0f660941d292 PLC500ED - 21:B0:1F	Edge Compute	17 de jan de 2023 10:07

Figura 3.6: Dispositivo conectado a la plataforma WEGnology.

3.1.6 Agregando Atributos

Los datos de telemetría del dispositivo son almacenados en forma de **Atributos**. Para crear nuevos atributos y, de esa forma, ser utilizados en las aplicaciones IoT, se debe hacer clic en **Devices > PLC500ED-XX:YY:ZZ > Attributes > Add**. Así que sea definido el nombre del atributo y el tipo de dato, haga clic en **Create Attribute**.

4 CODESYS - BIBLIOTECA WEGNOLOGY

La biblioteca WEGnology fue desarrollada para contribuir a la conectividad entre el software **Codesys**, plataforma **WEGnology** y container **Edge Agent**, responsable por el procesamiento de borde del PLC500ED. A través de los bloques de funciones y métodos disponibles, el intercambio de informaciones entre estas interfaces es realizado de forma simple, permitiendo la rápida configuración y utilización de las funcionalidades disponibles. La Figura 4.1 muestra la biblioteca WEGnology en el Codesys.

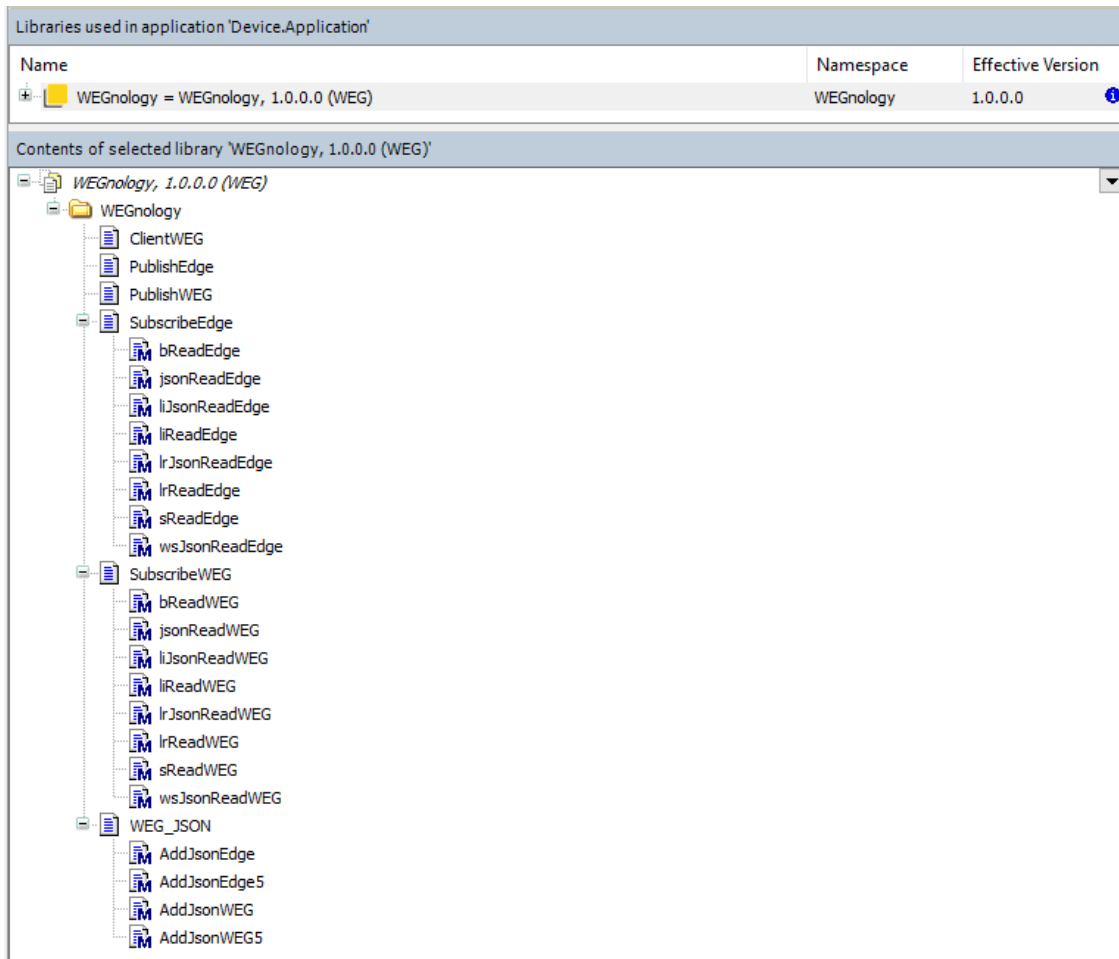


Figura 4.1: Biblioteca WEGnology.



¡NOTA!

La biblioteca **WEGnology** es instalada con el **WEG package**. Para más informaciones consulte el manual del PLC500 para más informaciones.



¡NOTA!

Para que la biblioteca **WEGnology** sea utilizada en el **Codesys** es necesaria, primeramente, la configuración del **panel de integración con la nube**, ubicado en la página web del producto.

4.1 RECOMENDACIONES

En el Codesys, se recomiendan los siguientes cuidados, implementando aplicaciones IoT:

- Utilización de tareas específicas para publicación y suscripción de payloads MQTT, cuando sea posible, y con baja prioridad (20 - 31), de forma de evitar interferencias en tareas críticas. Obs: para el Codesys, la tarea de mayor prioridad es la 0 (cero).
- No operar cargas críticas a través de plataformas IoT, ya que éstas pueden presentar una latencia elevada, además de la dependencia de una conexión estable con internet.
- Para mayores informaciones sobre los bloques de funciones y los métodos de la biblioteca WEGnology, consultar la ayuda de la propia biblioteca en el Codesys.

4.2 BIBLIOTECA CODESYS IIOT LIBRARIES SL

Para el funcionamiento de la biblioteca **WEGnology** es necesaria la instalación previa de la biblioteca **CODESYS IIoT Libraries SL**, la cual incluye otras diversas bibliotecas y protocolos, tales como:

- MQTT Client (MQTT)
- Web Client (http, https)
- AWS IoT Core Client (MQTT)
- Azure IoT Hub Client (MQTT, https)
- Mail Service (POP3, SMTP)
- SNMP Service (SNTP)
- SNMP Library (SNMP)
- SMS Service (SMS)
- JSON Utilities
- XML Utility

Para la instalación de esta biblioteca, abra el Codesys y acceda a **Tools > CODESYS Installer**. en la nueva ventana abierta, haga clic en **Browse** y busque "IIoT". Seleccione la biblioteca **IIoT Libraries SL** y haga clic en **Install**. Será necesario cerrar la aplicación del Codesys antes de finalizar la instalación. La Figura 4.2 muestra la ventana del CODESYS installer, a partir de donde la biblioteca CODESYS IIoT Libraries SL puede ser instalada.

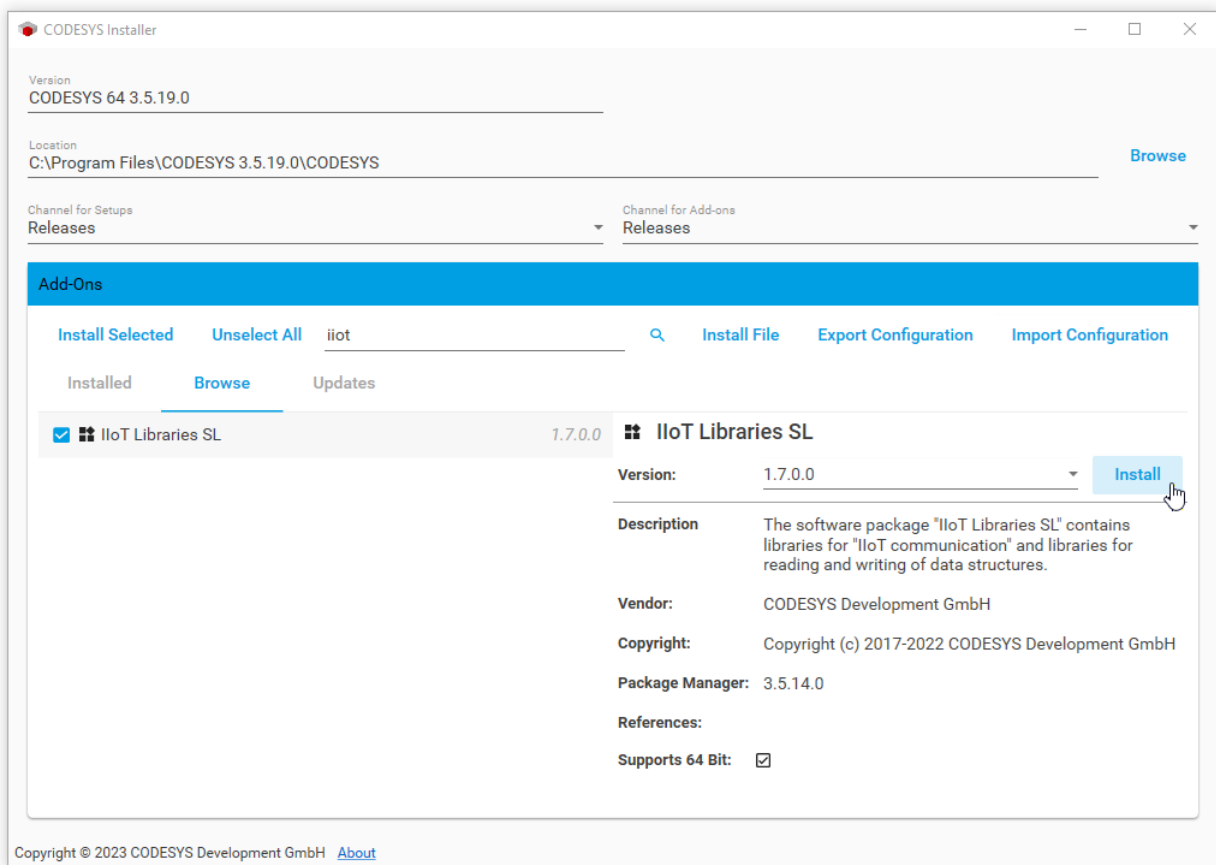


Figura 4.2: Instalación de la biblioteca CODESYS IIoT Libraries SL.

4.3 SINTAXIS DE LOS OBJETOS JSON PUBLICADOS

Los payloads MQTT publicados por el Codesys son ordenados en la forma de objetos JSON. Son utilizados estándares específicos de objetos JSON, de acuerdo con la forma de utilización de los datos, ya que pueden ser utilizados directamente por la plataforma WEGnology o por el procesamiento de flanco, donde diversos dispositivos pueden ser conectados al PLC500ED. Siguen los estándares de objetos JSON.

4.3.1 WEGnology

La plataforma WEGnology recibe los datos publicados en el siguiente formato:

```
{
  "data" :
  {
    "time" : "value",
    "attribute1" : "value1",
    "attribute2" : "value2",
    :
    "attributeN" : "valueN"
  }
}
```

En el Codesys, el usuario precisa definir solamente los nombres de los atributos y las variables de las cuales sus valores son obtenidos.

4.3.2 Edge Computing

El objeto JSON utilizado por el procesamiento de borde tiene un elemento específico para identificar el dispositivo de donde los datos son originados. En el ejemplo de abajo, se tiene un objeto JSON en el estándar utilizado por el procesamiento de flanco, siendo "device1" el nombre del dispositivo:

```
{
  "data" :
  {
    "time" : "value",
    "device1" :
    {
      "attribute1" : "value1",
      "attribute2" : "value2",
      :
      "attributeN" : "valueN"
    }
  }
}
```

Para el procesamiento de borde, el usuario precisa definir en el Codesys solamente el nombre del dispositivo, nombres de los atributos y las variables de las cuales sus valores son obtenidos.

Tópico de publicación: plc500ed/toAgent/state/<device_name>



¡NOTA!

Los nombres de los atributos publicados deben ser exactamente iguales a los registrados en la plataforma WEGnology, para que sean reconocidos adecuadamente.



¡NOTA!

El bloque de función **WEG_JSON**, junto a sus métodos, ordenan de forma automática los objetos JSON a ser publicados de acuerdo con la sintaxis correcta.



¡NOTA!

Los payloads destinados al procesamiento de borde deben ser tratados en la plataforma WEGnology para identificar el dispositivo de donde los datos son originados. Ej.: device1, device2, etc.

4.4 BLOQUES DE FUNCIONES Y MÉTODOS MQTT

Los bloques de funciones y los métodos que componen la biblioteca WEGnology son divididos de acuerdo con la forma de utilización de los datos. Éstos pueden ser utilizados directamente por la plataforma WEGnology – sufijo **WEG** – o por el procesamiento de borde (Edge Computing) – sufijo **Edge**. Abajo, la descripción de los bloques de función y los métodos disponibles. Para más detalles, consultar la documentación de la biblioteca WEGnology, directamente en el Codesys.

- **ClientWEG**: Conecta el dispositivo al broker interno, que a su vez se conecta a la plataforma WEGnology. Toda la aplicación IoT de la biblioteca WEGnology necesita un método ClientWEG, incluyendo las aplicaciones Edge. Solamente debe ser utilizada una instancia del ClientWEG para varios bloques de funciones de Publicar o Subscraver.



¡NOTA!

Se recomienda la utilización de tareas con intervalos de tiempo menores a **500 ms** para la ejecución del bloque **ClientWEG**.

- **PublishWEG**: Publica en el tópico MQTT estándar de la plataforma WEGnology. Es publicado el JSON creado por los métodos encontrados en el bloque de función WEG_JSON, sufijo WEG.
- **PublishEdge**: Publica en un tópico MQTT personalizado para ser utilizado por el procesamiento de borde. Es publicado el JSON creado por los métodos encontrados en el bloque de función WEG_JSON, sufijo Edge. Debe ser declarado solamente un PublishEdge para cada dispositivo conectado al PLC500ED.
- **SubscribeWEG**: Suscribe al tópico MQTT estándar de la plataforma WEGnology. Debe ser declarado solamente un SubscribeWEG. Los métodos específicos contenidos en el SubscribeWEG deben ser utilizados para cada tipo de variable. La tabla 4.1 presenta los métodos disponibles en el bloque de función SubscribeWEG.

Método	Descripción
bReadWEG	Recibe un comando del tipo BOOL de la plataforma y actualiza la variable de interés.
liReadWEG	Recibe un comando del tipo LINT de la plataforma y actualiza la variable de interés.
lrReadWEG	Recibe un comando del tipo LREAL de la plataforma y actualiza la variable de interés.
sReadWEG	Recibe un comando del tipo STRING de la plataforma y actualiza la variable de interés.
liJsonReadWEG	Recibe un objeto JSON de la plataforma y actualiza una variable del tipo LINT.
lrJsonReadWEG	Recibe un objeto JSON de la plataforma y actualiza una variable del tipo LREAL.
wsJsonReadWEG	Recibe un objeto JSON de la plataforma y actualiza una variable del tipo WSTRING.
jsonReadWEG	Recibe un objeto JSON de la plataforma y busca un Comando específico. Este método debe ser utilizado cuando el JSON recibido tiene una jerarquía de datos con varios niveles. El objeto JSON recibido también debe ser tratado para aislar las variables de interés.

Tabla 4.1: Métodos disponibles en el bloque de función SubscribeWEG.

- **SubscribeEdge**: Suscribe a un tópico MQTT personalizado para ser utilizado por el procesamiento de borde. Solamente debe ser declarado un SubscribeEdge para cada dispositivo conectado al PLC500ED. Deben ser utilizados métodos específicos contenidos en el SubscribeEdge para cada tipo de variable. La Tabla 4.2 muestra los métodos disponibles en el bloque de función SubscribeEdge.

Método	Descripción
bReadEdge	Recibe un comando del tipo BOOL y actualiza la variable de interés.
liReadEdge	Recibe un comando del tipo LINT y actualiza la variable de interés.
lrReadEdge	Recibe un comando del tipo LREAL y actualiza la variable de interés.
sReadEdge	Recibe un comando del tipo STRING y actualiza la variable de interés.
liJsonReadEdge	Recibe un objeto JSON y actualiza una variable del tipo LINT.
lrJsonReadEdge	Recibe un objeto JSON y actualiza una variable del tipo LREAL.
wsJsonReadEdge	Recibe un objeto JSON y actualiza una variable del tipo WSTRING.
jsonReadEdge	Recibe un objeto JSON y busca un Comando específico. Este método debe ser utilizado cuando el JSON recibido tiene una jerarquía de datos con varios niveles. El objeto JSON recibido también debe ser tratado para aislar las variables de interés.

Tabla 4.2: Métodos disponibles en el bloque de función SubscribeEdge.

- WEG_JSON**: Bloque de función que contiene métodos para agregar y actualizar atributos y valores al objeto JSON esperado por la plataforma WEGnology o por el Edge Agent. Los métodos de este bloque de función aceptan como entradas los siguientes tipos de variables: BOOL, BYTE, WORD, SINT, INT, UINT, DINT, UDINT, LINT, REAL y LREAL. Los datos son convertidos para WSTRING. Estos métodos pueden ser declarados diversas veces, agregando nuevas variables al JSON creado o actualizando los valores de las variables existentes. La Tabla 4.3 muestra los métodos disponibles en el bloque de funciones SubscribeEdge.

Método	Descripción
AddJsonWEG	Método para creación de un JSON, adición de un atributo y actualización de las variables referentes a la plataforma WEGnology.
AddJsonWEG5	Método para creación de un JSON, adición de cinco atributos y actualización de variables referentes a la plataforma WEGnology.
AddJsonEdge	Método para creación de un JSON, adición de atributos y actualización de variables referentes al procesamiento de borde.
AddJsonEdge5	Método para creación de un JSON, adición de cinco atributos y actualización de variables referentes al procesamiento de borde.

Tabla 4.3: Métodos disponibles en el bloque de funciones WEG_JSON.

4.5 EJEMPLOS DE APLICACIONES

En esta sección son presentados ejemplos de aplicaciones en el Codesys – en texto estructurado (ST) y en Ladder – y en la plataforma WEGnology para demostrar algunas funcionalidades IoT del PLC500ED. El intervalo de tiempo de las tareas es de 100 ms. Son mostrados los siguientes ejemplos:

- Publicación de payloads para la plataforma WEGnology
- Suscripción en tópicos de la plataforma WEGnology
- Publicación de payloads para el Edge Agent
- Suscripción en tópicos del Edge Agent

4.5.1 Publicación de payloads para la plataforma WEGnology

El ClientWEG es conectado al broker interno del PLC500ED, que a su vez se conecta a la plataforma WEGnology. Los valores de las variables del tipo BOOL, UINT y REAL son alteradas en cada ciclo y, cada 10 ciclos, sus valores son publicados en la plataforma bajo los atributos "temperature", "power" y "flag". En la Figura 4.3 es mostrada la declaración de las variables, y en la Figura 4.4 el payload recibido por la plataforma. El programa completo en ST es presentado en la Figura 4.5, y en Ladder, en la Figura 4.6.

```

PublishWEG_PRG
PROGRAM PublishWEG_PRG
VAR
  // CLIENT
  WEG_CLIENT : WEGnology.ClientWEG;
  xErrorClient : BOOL;
  eErrorClient : MQTT.MQTT_ERROR;
  xConnectedToBroker : BOOL;

  // PUBLISH WEG
  WEG_PUBWEG : WEGnology.PublishWEG;
  eErrorPubWEG : MQTT.MQTT_ERROR;
  xDonePubWEG, xErrorPubWEG, xBusyPubWEG : BOOL;

  // WEG_JSON
  WEG_JSON : WEGnology.WEG_JSON;
  xErrorAdd : BOOL;
  xDoneAdd1, xDoneAdd2, xDoneAdd3 : BOOL;

  // PRG
  rTemperature : REAL;
  uiPower, uiCount : UINT;
  bFlag, uiPub : BOOL;
  RS_1 : RS;
  TON_1 : TON;
END_VAR
  
```

Figura 4.3: Declaración de variables PublishWEG.

Device PLC500ED-21:B0:1F reported its state
MQTT wnology/63c82eb9cad5c854a76acd/state
qui 19 de jan de 2023 12:36:29.752 GMT-03:00

Received Payload
▼ (root) {} 1 key
▼ "data": {} 4 keys
"flag": "0"
"power": "662"
"temperature": "6.620057"
"time": "1674142589178"

Device PLC500ED-21:B0:1F reported its state
MQTT wnology/63c82eb9cad5c854a76acd/state
qui 19 de jan de 2023 12:36:27.851 GMT-03:00

Received Payload
▼ (root) {} 1 key
▼ "data": {} 4 keys
"flag": "1"
"power": "643"
"temperature": "6.430053"
"time": "1674142587278"

Device PLC500ED-21:B0:1F reported its state
MQTT wnology/63c82eb9cad5c854a76acd/state
qui 19 de jan de 2023 12:36:25.951 GMT-03:00

Figura 4.4: Payload PublishWEG recibido por la plataforma.

PublishWEG_PRG - Structured text (ST)

```

// Set Reset
RS_1(SET := NOT xErrorClient, RESET1 := xErrorClient);

// Timer
TON_1(IN := RS_1.Q1, PT := T#1S);

// FB ClientWEG
WEG_CLIENT(
  xEnable := TON_1.Q,
  xConnectedToBroker => xConnectedToBroker,
  xError => xErrorClient,
  eErrorType => eErrorClient);

// REAL variable
rTemperature := rTemperature + 0.01;
IF rTemperature > 100 THEN rTemperature := -100; END_IF;

// UINT variable
uiPower := uiPower + 1;
IF uiPower > 10000 THEN uiPower := 0; END_IF;

// BOOL variable
bFlag := NOT bFlag;

// Publish the payload each 10 x Task interval
IF uiPub = 0 THEN
  uiCount := uiCount + 1;
  IF uiCount > 10 THEN uiCount := 0; uiPub := 1; END_IF;
ELSE
  // Add the bFlag value to the "flag" attribute
  WEG_JSON.AddJsonWEG(
    xExecute := NOT xDoneAdd1,
    wsVarName := "flag", anyVar := bFlag,
    AddJsonWEG => xDoneAdd1);

  // Add the rTemperature value to the "temperature" attribute
  WEG_JSON.AddJsonWEG(
    xExecute := NOT xDoneAdd2,
    wsVarName := "temperature", anyVar := rTemperature,
    AddJsonWEG => xDoneAdd2);

  // Add the uiPower value to the "power" attribute
  WEG_JSON.AddJsonWEG(
    xExecute := NOT xDoneAdd3,
    wsVarName := "power", anyVar := uiPower,
    AddJsonWEG => xDoneAdd3);

  // Publish the created JSON object
  WEG_PUBWEG(
    xExecute := ((NOT xErrorPubWEG) AND (NOT xDonePubWEG) AND (xConnectedToBroker)),
    xDone => xDonePubWEG,
    xBusy => xBusyPubWEG,
    xError => xErrorPubWEG,
    eErrorType => eErrorPubWEG);

  IF (xDonePubWEG = 1 OR xErrorPubWEG = 1) THEN uiPub := 0; END_IF;
END_IF;

```

Figura 4.5: Programa PublishWEG en texto estructurado.

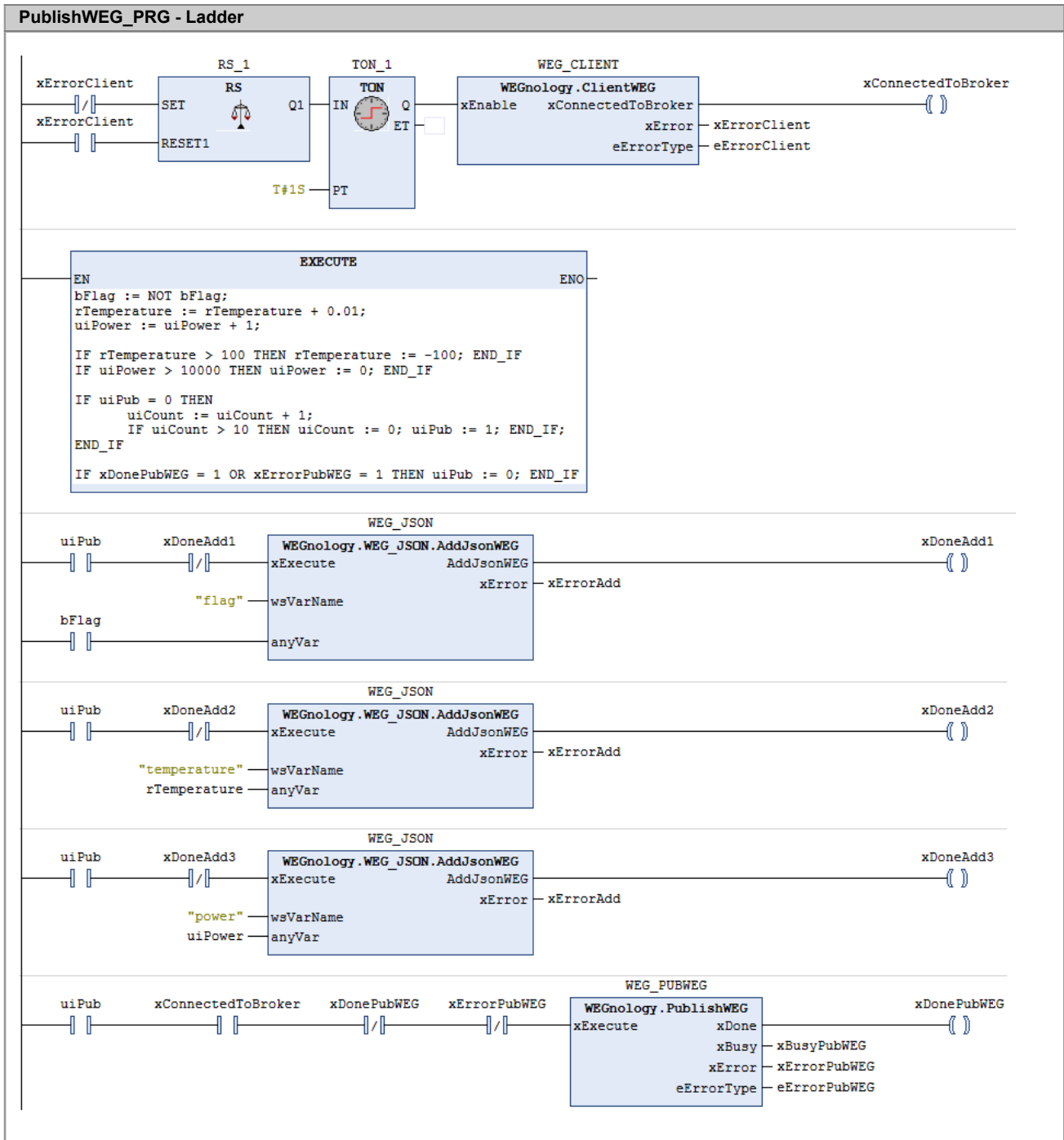


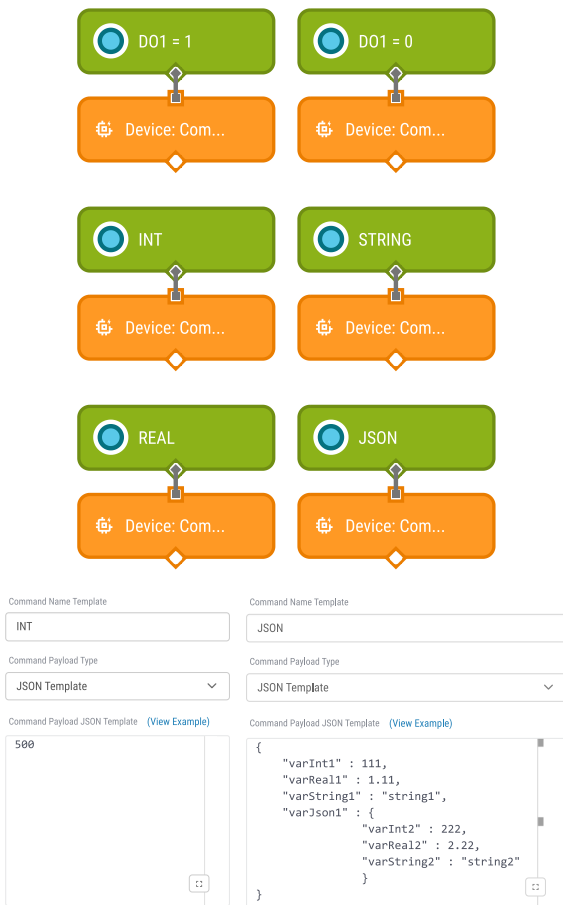
Figura 4.6: Programa PublishWEG en ladder.

4.5.2 Suscripción en tópicos de la plataforma WEGnology

Primeramente es creada una aplicación en la plataforma WEGnology para la publicación de los payloads que serán recibidos por el Codesys a través del bloque SubscribeWEG y de sus métodos. Para eso, en el panel principal de la plataforma, ir a **Workflows > Application Workflows** y crear la aplicación. En la Figura 4.7 es mostrado el workflow implementado para esta prueba, el cual utiliza los nodos **Virtual Button** y **Device: Command**.

Cuando es presionado un botón virtual, es publicado un comando MQTT con un **Command Name** específico, junto al **Command Payload** correspondiente. Como ejemplo, en la Figura 4.7 son mostrados comandos que envían un payload del tipo entero con el valor de 500, y uno que envía un objeto JSON.

En la aplicación del Codesys, el ClientWEG es conectado al broker interno del PLC500ED, que a su vez se conecta a la plataforma WEGnology. Cuando la flag **xSubscribeActive** está en estado alto, el bloque SubscribeWEG está activo y aguardando un payload. Las variables del tipo LINT y LREAL son incrementadas cada ciclo. Métodos del bloque SubscribeWEG aguardan payloads con atributos y tipos específicos. Cuando la plataforma envía algún comando con una variable válida, su valor es inmediatamente actualizado. La declaración de las variables utilizadas en el Codesys es mostrada en la Figura 4.8. En la Figura 4.9 se tiene el programa en texto estructurado, y en la Figura 4.10, el programa en Ladder.



```

SubscribeWEG_PRG

PROGRAM SubscribeWEG_PRG
VAR
  // CLIENT
  WEG_CLIENT : WEGnology.ClientWEG;
  xErrorClient : BOOL;
  eErrorClient : MQTT.MQTT_ERROR;
  xConnectedToBroker : BOOL;

  // SUBSCRIBE WEG
  WEG_SUBWEG : WEGnology.SubscribeWEG;
  eErrorSubWEG : MQTT.MQTT_ERROR;
  xErrorSubWEG, xBusySubWEG, xSubActiveWEG : BOOL;
  xReceivedSubWEG, xDonebSubWEG, xDoneliSubWEG : BOOL;
  xDonelrSubWEG, xDonesSubWEG : BOOL;
  xDoneliJsonSubWEG, xDonelrJsonSubWEG : BOOL;
  xDonesJsonSubWEG, xDoneJsonSubWEG : BOOL;

  // PRG
  bSubVar : BOOL;
  liSubVar, liJsonSubVar : LINT;
  lrSubVar, lrJsonSubVar : LREAL;
  sSubVar : STRING(JSON.GParams.g_diMaxStringSize);
  wsJsonSubVar : WSTRING(JSON.GParams.g_diMaxStringSize);
  wsPayloadJson : WSTRING(2048);
  udiSizeJson : UDINT;
  JSONDataJson : JSON.JSONData;
  RS_1 : RS;
  TON_1 : TON;
END_VAR
    
```

Figura 4.7: Workflow para envío de comandos SubscribeWEG.

Figura 4.8: Declaración de variables SubscribeWEG.

SubscribeWEG_PRG - Structured text (ST)

```

// Set Reset
RS_1(SET := NOT xErrorClient, RESET1 := xErrorClient);

// Timer
TON_1(IN := RS_1.Q1, PT := T#1S);

// FB ClientWEG
WEG_CLIENT(
  xEnable := TON_1.Q,
  xConnectedToBroker => xConnectedToBroker,
  xError => xErrorClient,
  eErrorType => eErrorClient);

// LINT and LREAL variables
liSubVar := liSubVar + 1;
lrSubVar := lrSubVar + 0.01;
liJsonSubVar := liJsonSubVar + 1;
lrJsonSubVar := lrJsonSubVar + 0.01;

IF liSubVar > 1000 THEN liSubVar := 0; END_IF
IF liJsonSubVar > 1000 THEN liJsonSubVar := 0; END_IF
IF lrSubVar > 1000 THEN lrSubVar := -1000; END_IF
IF lrJsonSubVar > 1000 THEN lrJsonSubVar := -1000; END_IF

// Subscribe to the WEGnology platform
WEG_SUBWEG(
  xEnable := (NOT xErrorSubWEG) AND (NOT xReceivedSubWEG),
  xReceived => xReceivedSubWEG,
  xBusy => xBusySubWEG,
  xError => xErrorSubWEG,
  eErrorType => eErrorSubWEG,
  xSubscribeActive => xSubActiveWEG);

// Methods to receive a Command Name called "sVarName" with different types of payloads (BOOL, LINT, LREAL or STRING)
WEG_SUBWEG.bReadWEG(
  xExecute := NOT xDonebSubWEG, sVarName := 'DO1', bVar := bSubVar, bReadWEG => xDonebSubWEG);

WEG_SUBWEG.liReadWEG(
  xExecute := NOT xDoneliSubWEG, sVarName := 'INT', liVar := liSubVar, liReadWEG => xDoneliSubWEG);

WEG_SUBWEG.lrReadWEG(
  xExecute := NOT xDonelrSubWEG, sVarName := 'REAL', lrVar := lrSubVar, lrReadWEG => xDonelrSubWEG);

WEG_SUBWEG.sReadWEG(
  xExecute := NOT xDonesSubWEG, sVarName := 'STRING', sVar := sSubVar, sReadWEG => xDonesSubWEG);

// Methods to receive an object JSON, which Command Name is called "JSON", and it searches for a wsKey of different types (LINT, LREAL or WSTRING)
WEG_SUBWEG.liJsonReadWEG(xExecute := NOT xDoneliJsonSubWEG,
  wsCommandName := "JSON", wsKey := "varInt1", liVar := liJsonSubVar, liJsonReadWEG => xDoneliJsonSubWEG);

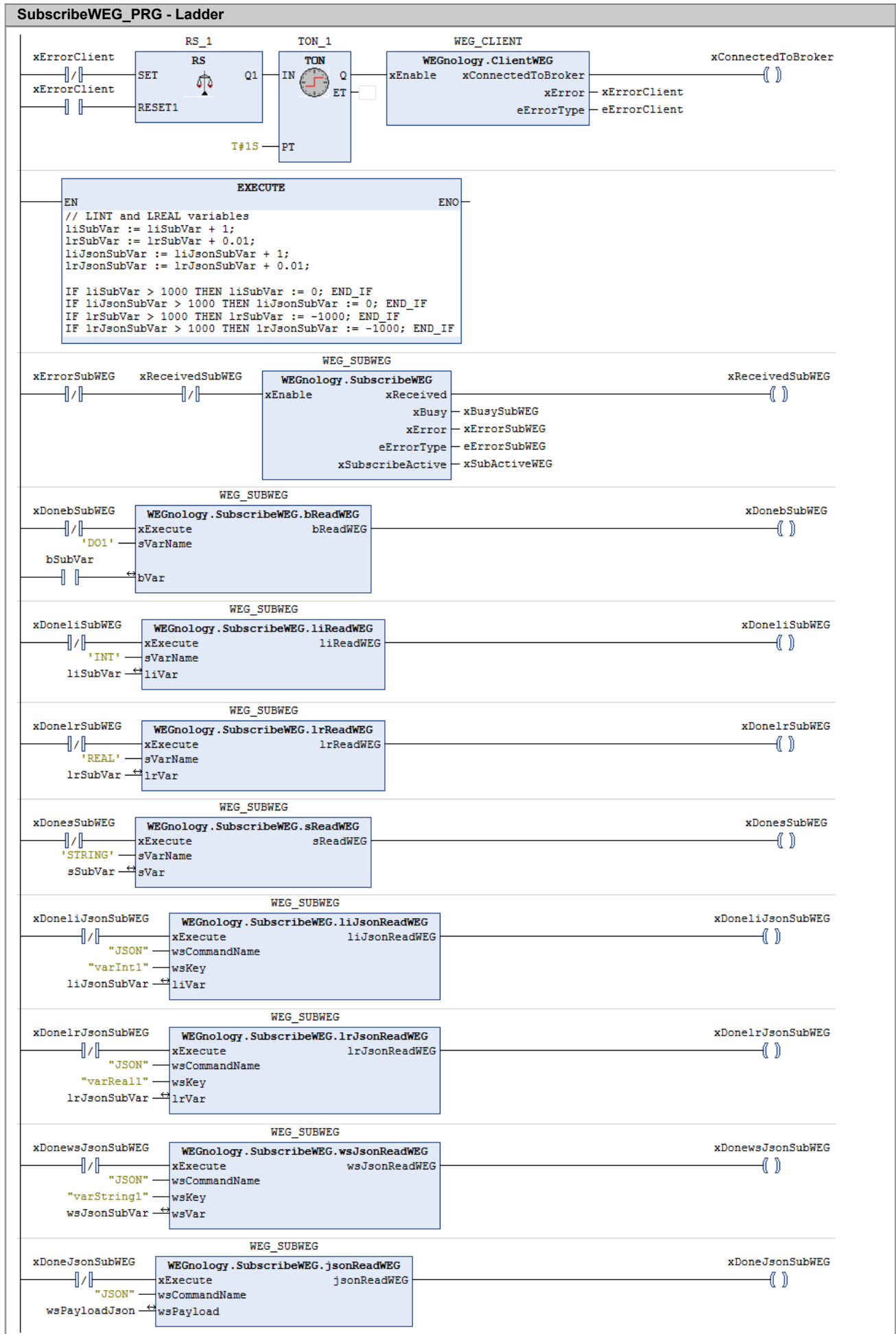
WEG_SUBWEG.lrJsonReadWEG(xExecute := NOT xDonelrJsonSubWEG,
  wsCommandName := "JSON", wsKey := "varReal1", lrVar := lrJsonSubVar, lrJsonReadWEG => xDonelrJsonSubWEG);

WEG_SUBWEG.wsJsonReadWEG(xExecute := NOT xDonewsJsonSubWEG,
  wsCommandName := "JSON", wsKey := "varString1", wsVar := wsJsonSubVar, wsJsonReadWEG => xDonewsJsonSubWEG);

// Methods to receive an object JSON. Its Command Name is called "JSON" and its payload must be treated.
WEG_SUBWEG.jsonReadWEG(xExecute := NOT xDoneJsonSubWEG,
  wsCommandName := "JSON", wsPayload := wsPayloadJson, jsonReadWEG => xDoneJsonSubWEG);

```

Figura 4.9: Programa SubscribeWEG en texto estructurado.



4.5.3 Publicación de payloads para el Edge Agent

Con la utilización del bloque de función PublishEdge es posible publicar payloads de diferentes dispositivos conectados al PLC500ED. Los payloads son publicados en los tópicos **plc500ed/toAgent/state/<device_name>**, onde <device_name> é o nome do dispositivo de qual as variáveis são provenientes. É necessário definir um ponteiro para JSON para cada dispositivo conectado ao PLC, como pode ser visto na Figura 4.11, que muestra la declaración de las variables en el Codesys.

De forma semejante a las demás aplicaciones, el bloque ClientWEG es utilizado para conectar el PLC500ED al broker interno. Las variables del tipo REAL, UINT Y BOOL son alteradas cada ciclo, simulando una aplicación compleja. Es creado un primer objeto JSON, proveniente del **device1**, con a las variables rTemperature1, uiPower1 y bFlag1, que son publicadas bajo los atributos "temperature", "power" y "flag". El segundo JSON es referente al **device2**, y está compuesto por la variable rTemperature2, publicada también bajo el atributo "temperature". El último JSON es formado por la variable uiPower3, enviada bajo el atributo "power", al **device3**.



¡NOTA!

Para la utilización de las funcionalidades de procesamiento de borde, la imagen del **Edge Agent** debe ser habilitada en la página web del producto, en **Página de Configuración > Docker > On**.



¡NOTA!

Se recomienda utilizar un enclavamiento entre el PublisherEdge y los métodos AddJsonEdge, ya que eso impide que el objeto JSON sea modificado mientras el publisher intenta enviarlo.

PublishEdge_PRG

```
PROGRAM PublishEdge_PRG
```

```
VAR
```

```
  // CLIENT
```

```
  WEG_CLIENT : WEGnology.ClientWEG;  
  xErrorClient : BOOL;  
  eErrorClient : MQTT.MQTT_ERROR;  
  xConnectedToBroker : BOOL;
```

```
  // PUBLISH EDGE
```

```
  WEG_PUBEDGE1, WEG_PUBEDGE2, WEG_PUBEDGE3 : WEGnology.PublishEdge;  
  xDonePubEdge1, xErrorPubEdge1, xBusyPubEdge1 : BOOL;  
  xDonePubEdge2, xErrorPubEdge2, xBusyPubEdge2 : BOOL;  
  xDonePubEdge3, xErrorPubEdge3, xBusyPubEdge3 : BOOL;  
  eErrorPubEdge1, eErrorPubEdge2, eErrorPubEdge3 : MQTT.MQTT_ERROR;
```

```
  // POINTER TO JSON - SEVERAL DEVICES
```

```
  JSONDataFactory1, JSONDataFactory2, JSONDataFactory3 : JSON.JSONDataFactory;  
  eFactoryError1, eFactoryError2, eFactoryError3 : FBF.ERROR;  
  pJsonData1 : POINTER TO JSON.JSONData := JSONDataFactory1.Create(eError => eFactoryError1);  
  pJsonData2 : POINTER TO JSON.JSONData := JSONDataFactory2.Create(eError => eFactoryError2);  
  pJsonData3 : POINTER TO JSON.JSONData := JSONDataFactory3.Create(eError => eFactoryError3);
```

```
  // WEG_JSON
```

```
  WEG_JSON : WEGnology.WEG_JSON;  
  xDoneAdd1, xDoneAdd2, xDoneAdd3 : BOOL;  
  xDoneAdd4, xDoneAdd5, xErrorAdd : BOOL;
```

```
  // PRG
```

```
  rTemperature1, rTemperature2 : REAL;  
  uiPower1, uiPower3 : UINT;  
  bFlag1 : BOOL := 0;  
  uiCount : UINT := 0;  
  RS_1 : RS;  
  TON_1 : TON;  
  timer : BLINK; // Blinking signal (turning on and off for specific durations)
```

```
END_VAR
```

Figura 4.11: Declaración de variables PublishEdge.

En la plataforma WEGnology es creada una aplicación para recibir payloads publicados en los tópicos "plc500ed/toAgent/state/device1", "plc500ed/toAgent/state/device2" y "plc500ed/toAgent/state/device3" a través del nodo **trigger MQTT**. Cuando algo es recibido en alguno de estos tópicos, el objeto JSON integrante del payload es decodificado a través del nodo **JSON: Decode**, y el resultado es mostrado por el node de **Debug**. En la Figura 4.12 es mostrado el workflow creado en la plataforma WEGnology y en la Figura 4.13 se tienen los payloads recibidos de los tres dispositivos, a través de esta aplicación ejemplo.

El programa completo de la aplicación PublishEdge del Codesys, en texto estructurado, es mostrado en la Figura 4.14, y el respectivo programa en Ladder, es presentado en la Figura 4.15.

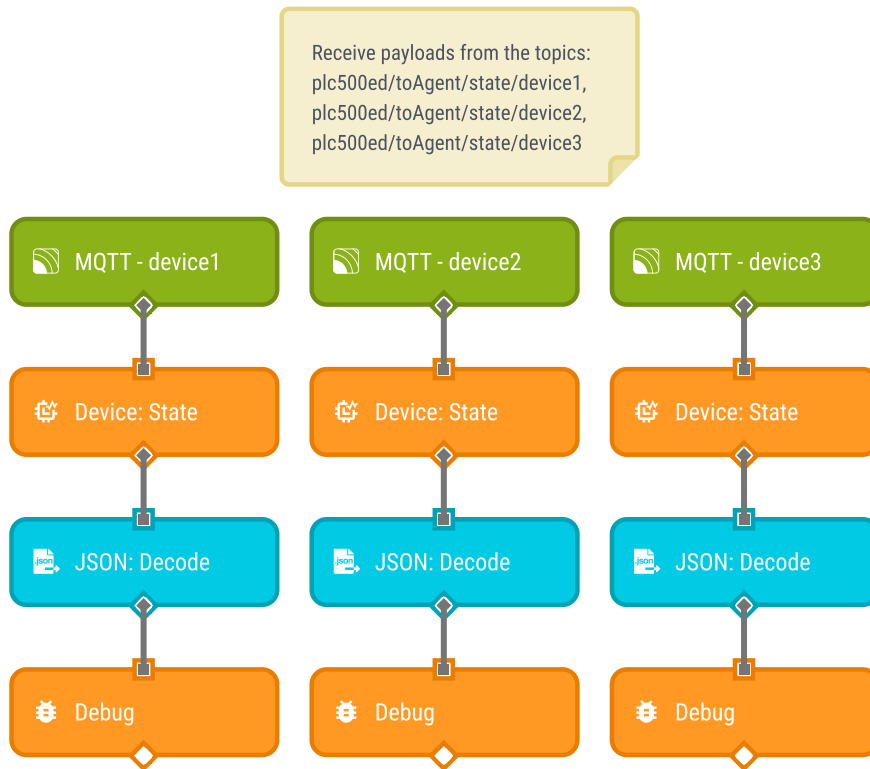


Figura 4.12: Workflow para recepción de payloads PublishEdge.

PLC500ED-21:B0:1F / 2023-01-23T19-19-22 / Debug (12ms) Debug Node Output seg 23 de jan de 2023 16:20:46.743 GMT-03:00	PLC500ED-21:B0:1F / 2023-01-23T19-01-47 / Debug (31ms) Debug Node Output seg 23 de jan de 2023 16:14:56.927 GMT-03:00	PLC500ED-21:B0:1F / 2023-01-23T19-01-47 / Debug (15ms) Debug Node Output seg 23 de jan de 2023 16:14:56.931 GMT-03:00
<pre>▼ data {} 1 key ▼ "data": {} 2 keys ▼ "device1": {} 3 keys "flag": "0" "power": "399" "temperature": "3.989997" "time": "1674501646667"</pre>	<pre>▼ data {} 1 key ▼ "data": {} 2 keys ▼ "device2": {} 1 key "temperature": "21.02039" "time": "1674501293567"</pre>	<pre>▼ data {} 1 key ▼ "data": {} 2 keys ▼ "device3": {} 1 key "power": "2102" "time": "1674501293568"</pre>

Figura 4.13: Payloads recibidos PublishEdge.

PublishEdge_PRG - Structured text (ST)

```

// Set Reset
RS_1(SET := NOT xErrorClient, RESET1 := xErrorClient);

// Timer
TON_1(IN := RS_1.Q1, PT := T#1S);

// FB ClientWEG
WEG_CLIENT( xEnable := TON_1.Q,
  xConnectedToBroker => xConnectedToBroker, xError => xErrorClient, eErrorType => eErrorClient);

// Increment of variables and limitations
rTemperature1 := rTemperature1 + 0.01; rTemperature2 := rTemperature2 + 0.01;
uiPower1 := uiPower1 + 1; uiPower3 := uiPower3 + 1; bFlag1 := NOT bFlag1;
IF rTemperature1 > 100 THEN rTemperature1 := -100; END_IF
IF rTemperature2 > 100 THEN rTemperature2 := -100; END_IF
IF uiPower1 > 10000 THEN uiPower1 := 0; END_IF
IF uiPower3 > 10000 THEN uiPower3 := 0; END_IF

// Boolean square wave (2s off, 2s on)
timer(ENABLE := 1, TIMELOW := T#2s, TIMEHIGH := T#2s);

// Add the bFlag1 value to the "flag" attribute of the "device1"
WEG_JSON.AddJsonEdge(
  xExecute := NOT xDoneAdd1, AddJsonEdge => xDoneAdd1, wsVarName := "flag", anyVar := bFlag1,
  wsDeviceName := "device1", pJsonData := pJsonData1);

// Add the rTemperature1 value to the "temperature" attribute of the "device1"
WEG_JSON.AddJsonEdge(
  xExecute := NOT xDoneAdd2, AddJsonEdge => xDoneAdd2, wsVarName := "temperature", anyVar := rTemperature1,
  wsDeviceName := "device1", pJsonData := pJsonData1);

// Add the uiPower1 value to the "power" attribute of the "device1"
WEG_JSON.AddJsonEdge(
  xExecute := NOT xDoneAdd3, AddJsonEdge => xDoneAdd3, wsVarName := "power", anyVar := uiPower1,
  wsDeviceName := "device1", pJsonData := pJsonData1);

// Add the rTemperature2 value to the "temperature" attribute of the "device2"
WEG_JSON.AddJsonEdge(
  xExecute := NOT xDoneAdd4, AddJsonEdge => xDoneAdd4, wsVarName := "temperature", anyVar := rTemperature2,
  wsDeviceName := "device2", pJsonData := pJsonData2);

// Add the uiPower3 value to the "power" attribute of the "device3"
WEG_JSON.AddJsonEdge(
  xExecute := NOT xDoneAdd5, AddJsonEdge => xDoneAdd5, wsVarName := "power", anyVar := uiPower3,
  wsDeviceName := "device3", pJsonData := pJsonData3);

// Publish the created JSON object of the "device1"
WEG_PUBEDGE1(
  xExecute := ((timer.OUT) AND (NOT xErrorPubEdge1) AND (NOT xDonePubEdge1) AND (xConnectedToBroker)),
  xDone => xDonePubEdge1, xBusy => xBusyPubEdge1, xError => xErrorPubEdge1, eErrorType => eErrorPubEdge1,
  pJsonData := pJsonData1);

// Publish the created JSON object of the "device2"
WEG_PUBEDGE2(
  xExecute := ((timer.OUT) AND (NOT xErrorPubEdge2) AND (NOT xDonePubEdge2) AND (xConnectedToBroker)),
  xDone => xDonePubEdge2, xBusy => xBusyPubEdge2, xError => xErrorPubEdge2, eErrorType => eErrorPubEdge2,
  pJsonData := pJsonData2);

// Publish the created JSON object of the "device3"
WEG_PUBEDGE3(
  xExecute := ((timer.OUT) AND (NOT xErrorPubEdge3) AND (NOT xDonePubEdge3) AND (xConnectedToBroker)),
  xDone => xDonePubEdge3, xBusy => xBusyPubEdge3, xError => xErrorPubEdge3, eErrorType => eErrorPubEdge3,
  pJsonData := pJsonData3);

```

Figura 4.14: Programa PublishEdge en texto estructurado.

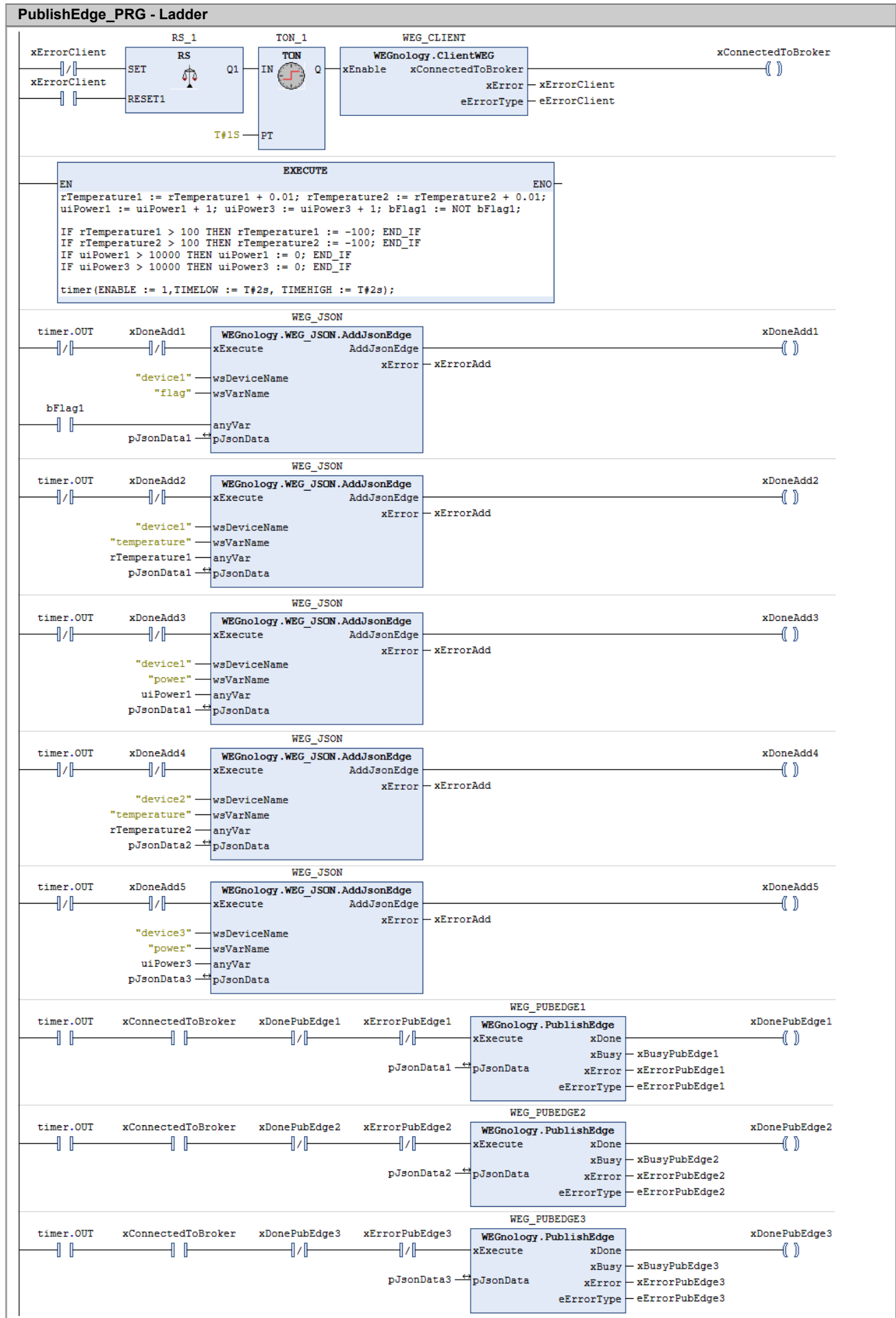


Figura 4.15: Programa PublishEdge en ladder.

4.5.4 Suscripción en tópicos del Edge Agent

Con la utilización del bloque SubscribeEdge es posible suscribir a diferentes tópicos MQTT, desde que tengan la sintaxis **plc500ed/fromAgent/command/<device_name>**, donde <device_name> es el nombre del dispositivo del cual son provenientes las variables. Son ofrecidos métodos para manipular variables de diferentes tipos.

Inicialmente, es creada una aplicación en la plataforma WEGnology para publicar payloads en tres tópicos específicos, representando tres diferentes dispositivos, a través de los nodos de salida MQTT. Es utilizado el trigger MQTT para fines de debug, a través del cual es posible visualizar el payload enviado. La Figura 4.17 muestra el workflow implementado en la plataforma.



Figura 4.16: Workflow para envío de los payloads SubscribeEdge.

En la Figura 4.17 son presentados los payloads del tipo entero y del tipo objeto JSON configurados para publicación por el workflow. Los demás nodos de salida MQTT siguen la misma estructura. La Figura 4.17(a) muestra el contenido de los nodos MQTT y la Figura 4.17(b) el debug realizado por la plataforma.

Broker: WEGnology Broker

Topic Template: plc500ed/fromAgent/command/device1

Message Template:

```
{
  "name": "INT",
  "payload": 100
}
```

Broker: WEGnology Broker

Topic Template: plc500ed/fromAgent/command/device1

Message Template:

```
{
  "name": "JSON",
  "payload": {
    "INT": 789,
    "REAL": 7.89
  }
}
```

(a)

(b)

Figura 4.17: Payloads publicados por el workflow SubscribeEdge.

En la aplicación del Codesys, el bloque ClientWEG es utilizado para conectar el PLC500ED al broker interno. Los bloques SubscribeEdge se suscriben en los tópicos estándar referentes a cada dispositivo específico. Los métodos aguardan la llegada de los payloads y verifican si el nombre del atributo es válido. Si lo es, el valor de la variable es actualizado. En los ejemplos en cuestión son mostrados métodos para recibir variables del tipo BOOL, LINT, LREAL y objeto JSON. La Figura 4.18 presenta la declaración de las variables de la aplicación SubscribeEdge.

El programa completo de la aplicación SubscribeEdge del Codesys, en texto estructurado, es mostrado en la Figura 4.20, y el respectivo programa en Ladder es presentado en la Figura 4.21. En la Figura 4.19 es mostrada la aplicación SubscribeEdge en funcionamiento, recibiendo un JSON en el tipo WSTRING.



¡NOTA!

Para la utilización de las funcionalidades de procesamiento de borde, la imagen del **Edge Agent** debe ser habilitada en la página web del producto, en **Página de Configuración > Docker > On**.

```

SubscribeEdge_PRG

PROGRAM SubscribeEdge_PRG
VAR
  // CLIENT
  WEG_CLIENT : WEGnology.ClientWEG;
  xErrorClient : BOOL;
  eErrorClient : MQTT.MQTT_ERROR;
  xConnectedToBroker : BOOL;
  RS_1 : RS;
  TON_1 : TON;

  // SUBSCRIBE EDGE
  WEG_SUBEDGE1, WEG_SUBEDGE2, WEG_SUBEDGE3 : WEGnology.SubscribeEdge;
  xErrorSubEdge1, xBusySubEdge1, xSubActiveEdge1, xReceivedSubEdge1 : BOOL;
  xErrorSubEdge2, xBusySubEdge2, xSubActiveEdge2, xReceivedSubEdge2 : BOOL;
  xErrorSubEdge3, xBusySubEdge3, xSubActiveEdge3, xReceivedSubEdge3 : BOOL;
  eErrorSubEdge1, eErrorSubEdge2, eErrorSubEdge3 : MQTT.MQTT_ERROR;
  xDonebSubEdge1, xDoneliSubEdge1, xDonelrSubEdge1, xDoneJsonSubEdge1 : BOOL;
  xDoneliSubEdge2, xDonelrSubEdge3 : BOOL;
  wsPayloadSubEdge1 : WSTRING(2048);

  // PGR
  bSubVar1 : BOOL;
  liSubVar1, liSubVar2 : LINT;
  lrSubVar1, lrSubVar3 : LREAL;
END_VAR
    
```

Figura 4.18: Declaración de variables SubscribeEdge.

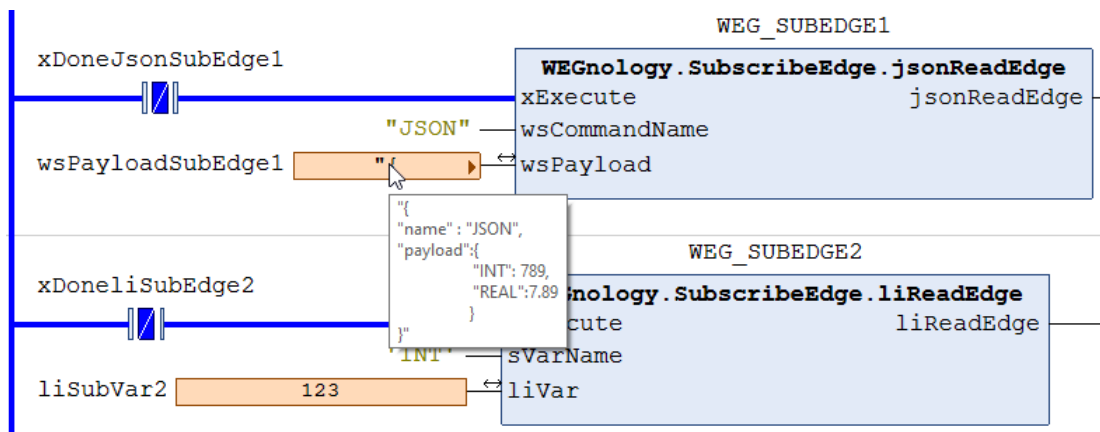


Figura 4.19: Aplicación SubscribeEdge recibiendo un JSON en el tipo WSTRING.

SubscribeEdge_PRG - Structured text (ST)

```

// Set Reset
RS_1(SET := NOT xErrorClient, RESET1 := xErrorClient);

// Timer
TON_1(IN := RS_1.Q1, PT := T#1S);

// FB ClientWEG
WEG_CLIENT( xEnable := TON_1.Q,
  xConnectedToBroker => xConnectedToBroker, xError => xErrorClient, eErrorType => eErrorClient);

// Subscribe to the "device1"
WEG_SUBEDGE1( xEnable := (NOT xErrorSubEdge1) AND (NOT xReceivedSubEdge1),
  xBusy => xBusySubEdge1, XError => xErrorSubEdge1, eErrorType => eErrorSubEdge1,
  xReceived => xReceivedSubEdge1, xSubscribeActive => xSubActiveEdge1,
  wsDeviceName := "device1");

// Subscribe to the "device2"
WEG_SUBEDGE2( xEnable := (NOT xErrorSubEdge2) AND (NOT xReceivedSubEdge2),
  xBusy => xBusySubEdge2, XError => xErrorSubEdge2, eErrorType => eErrorSubEdge2,
  xReceived => xReceivedSubEdge2, xSubscribeActive => xSubActiveEdge2,
  wsDeviceName := "device2");

// Subscribe to the "device3"
WEG_SUBEDGE3( xEnable := (NOT xErrorSubEdge3) AND (NOT xReceivedSubEdge3),
  xBusy => xBusySubEdge3, XError => xErrorSubEdge3, eErrorType => eErrorSubEdge3,
  xReceived => xReceivedSubEdge3, xSubscribeActive => xSubActiveEdge3,
  wsDeviceName := "device3");

// Methods to receive attributes with different types of payloads (BOOL, LINT, LREAL and JSON) from the device1
WEG_SUBEDGE1.bReadEdge(
  xExecute := NOT xDonebSubEdge1, sVarName := 'DO1', bVar := bSubVar1, bReadEdge => xDonebSubEdge1);

WEG_SUBEDGE1.liReadEdge(
  xExecute := NOT xDoneliSubEdge1, sVarName := 'INT', liVar := liSubVar1, liReadEdge => xDoneliSubEdge1);

WEG_SUBEDGE1.lrReadEdge(
  xExecute := NOT xDonelrSubEdge1, sVarName := 'REAL', lrVar := lrSubVar1, lrReadEdge => xDonelrSubEdge1);

WEG_SUBEDGE1.jsonReadEdge(
  xExecute := NOT xDoneJsonSubEdge1, wsCommandName := "JSON", wsPayload := wsPayloadSubEdge1, jsonReadEdge =>
xDoneJsonSubEdge1);

// Method to receive an attribute of the LINT type from the device2
WEG_SUBEDGE2.liReadEdge(
  xExecute := NOT xDoneliSubEdge2, sVarName := 'INT', liVar := liSubVar2, liReadEdge => xDoneliSubEdge2);

// Method to receive an attribute of the LREAL type from the device3
WEG_SUBEDGE3.lrReadEdge(
  xExecute := NOT xDonelrSubEdge3, sVarName := 'REAL', lrVar := lrSubVar3, lrReadEdge => xDonelrSubEdge3);

```

Figura 4.20: Programa SubscribeEdge en texto estructurado.

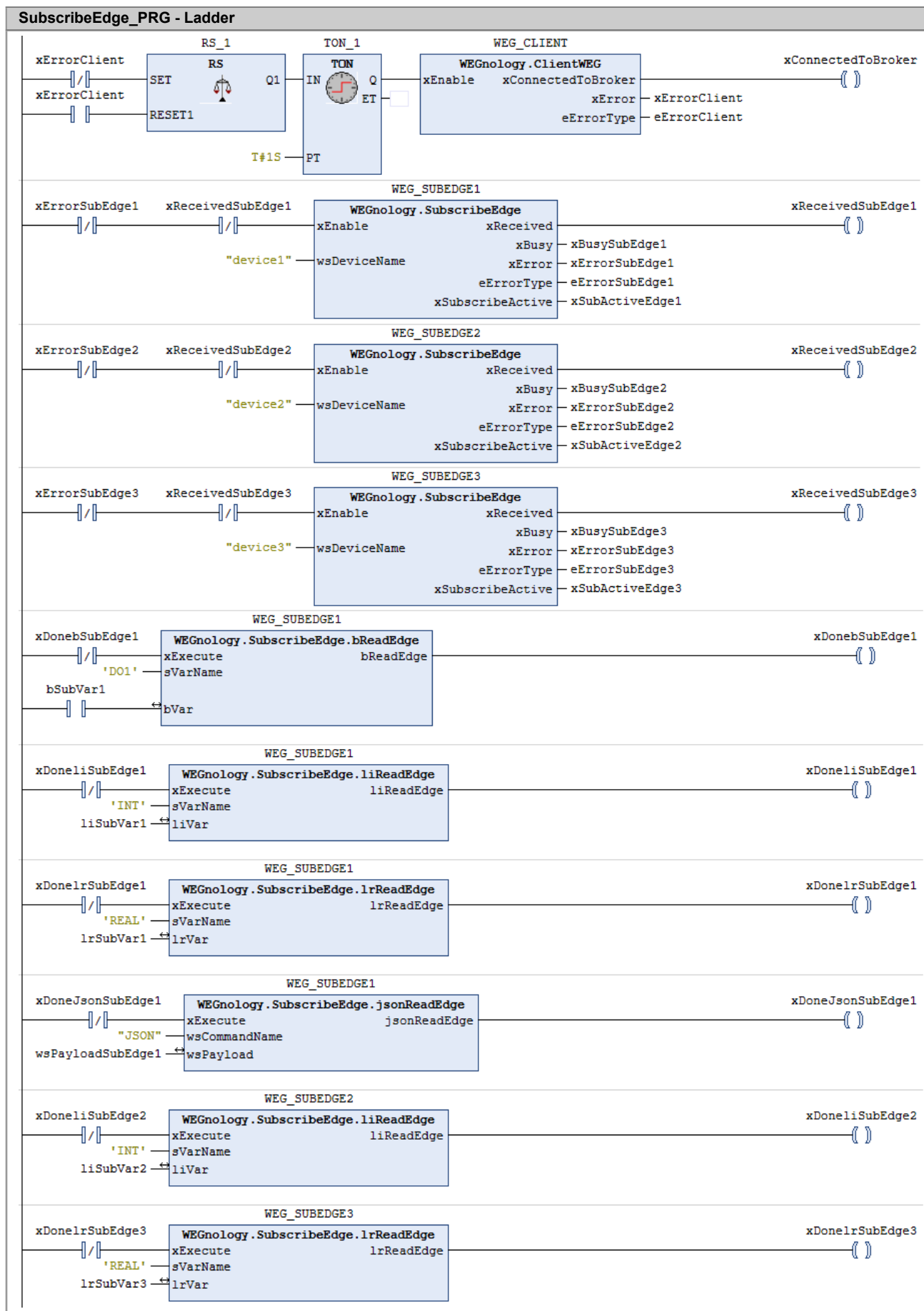


Figura 4.21: Programa SubscribeEdge en ladder.

5 GUÍA DE INICIO RÁPIDO

Esta sección muestra brevemente cómo se pueden realizar las configuraciones básicas del producto para establecer una conexión con la plataforma WEGnology y así ejecutar una primera aplicación IoT. Para más detalles, consulte el resto de esta Nota de Aplicación y el Manual del Usuario del PLC500, disponibles en www.weg.net.

El inicio rápido se puede realizar básicamente en cinco pasos:

5.1 ESTABLECER CONEXIÓN CON EL PRODUCTO

Para acceder al dispositivo, ya sea a través de la página web o de Codesys, el usuario debe colocar la computadora o el dispositivo móvil en la misma red que el producto. El primer acceso se puede hacer vía USB o ETH. Aquí están las direcciones IP iniciales de las interfaces:

Tabla 5.1: Dirección de IP estándar.

Conexión	Dirección de IP estándar
ETH1	192.168.1.10
ETH2	192.168.2.10
USB2	192.168.234.234

5.2 CONECTE EL PRODUCTO A INTERNET A TRAVÉS DE CODESYS O LA PÁGINA WEB

Con el dispositivo conectado a la misma red que el producto, el usuario puede configurar el acceso a Internet a través de la página web o Codesys.

- **Codesys:** a través de la pestaña **Configuración**, configure el ETH respectivo como **DHCP** y descargue la aplicación.
- **Página web:** En la **Página de configuración** de la página web, configure el ETH respectivo. Marque **Yes** en la opción **DHCP** y en **Default Route**.

Después de completar todos los campos, guarde la configuración haciendo clic en el botón **Save Configuration**. Se abrirá una ventana indicando que se reiniciará la aplicación. Espere hasta que finalice el proceso y vuelva a aparecer la pantalla de inicio de sesión.

La conexión a Internet se ha establecido cuando el indicador **Internet Status** aparece como **Connected**, en **Página de estado > Información del sistema**, como se muestra en la Figura 5.1.

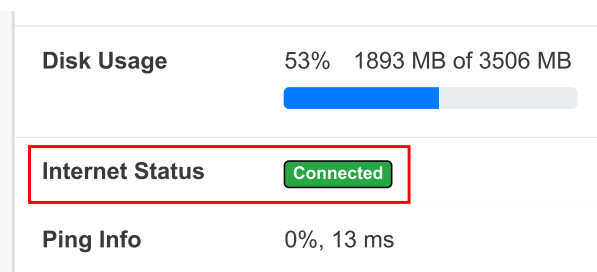


Figura 5.1: Conexão com a internet realizada com sucesso.

5.3 CONFIGURAR LA APLICACIÓN IOT A TRAVÉS DE LA PÁGINA WEB

Para realizar la configuración de la aplicación IoT, es necesario que la conexión a internet esté activa, como se muestra en el ítem anterior. Además, la aplicación IoT debe estar previamente creada en la plataforma [WEGnology](#).

La configuración de las credenciales se realiza en la **Página de configuración > Panel de integración en la nube** de la página web. Hay dos formas de configurar las credenciales de acceso:

- **Finalización manual:** complete los cinco campos que se describen a continuación:
 - **Integrador:** WEGnology v1 - servicio de integración en la nube.
 - **Application ID:** ID de una aplicación existente en WEGnology.
 - **Token de API:** Token de permiso de administración de aplicaciones
 - **Access Key:** Clave de autenticación para establecer la conexión del dispositivo en la aplicación WEGnology.
 - **Secret:** Contraseña de autenticación para establecer la conexión del dispositivo en la aplicación WEGnology.
- **Upload de archivos:** la información de **Token** y **Access Keys** se puede cargar a través de archivos .txt que contienen dichas credenciales. Para hacerlo, simplemente haga clic en **Choose File** y seleccione los archivos correspondientes (que deben descargarse al crear la aplicación IoT).

Con la configuración realizada, haga clic en el botón **Save Configuration** para guardar y reiniciar la aplicación. Después de completar este proceso, la pantalla de inicio de sesión se cargará nuevamente. El estado de la configuración de conexión de la plataforma se puede verificar en **MQTT Configuration Statu**, como se muestra en la Figura 5.2.

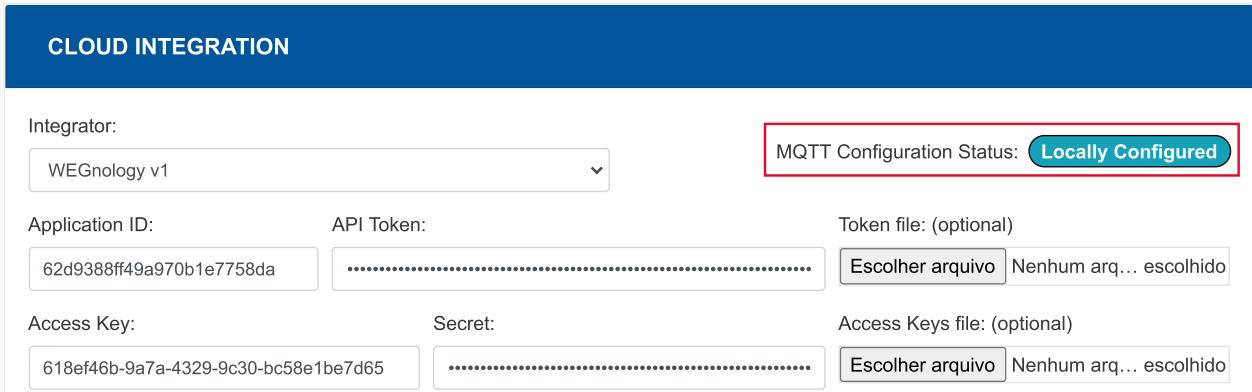


Figura 5.2: Conectado con éxito a la plataforma WEGnology.

5.4 HABILITAR UNA IMAGEN DOCKER

En la pestaña **Página de configuración > Docker** es posible habilitar/deshabilitar imágenes según las necesidades del usuario. Para hacerlo, simplemente cambie el **Status** entre **On** y **Off**, y luego haga clic en el botón **Save Configuration** para guardar y reiniciar la aplicación.

En la versión de firmware **1.2.0** del PLC500ED y **2.3.0** de Coreapp, el contenedor **Edge Agent** versión **1.34.0** está disponible de forma predeterminada.

La figura 5.3 muestra la Página de Estado del producto, donde puede ver que el dispositivo está conectado a Internet, con una aplicación de IoT configurada y con la imagen del contenedor de Edge Agent activa.

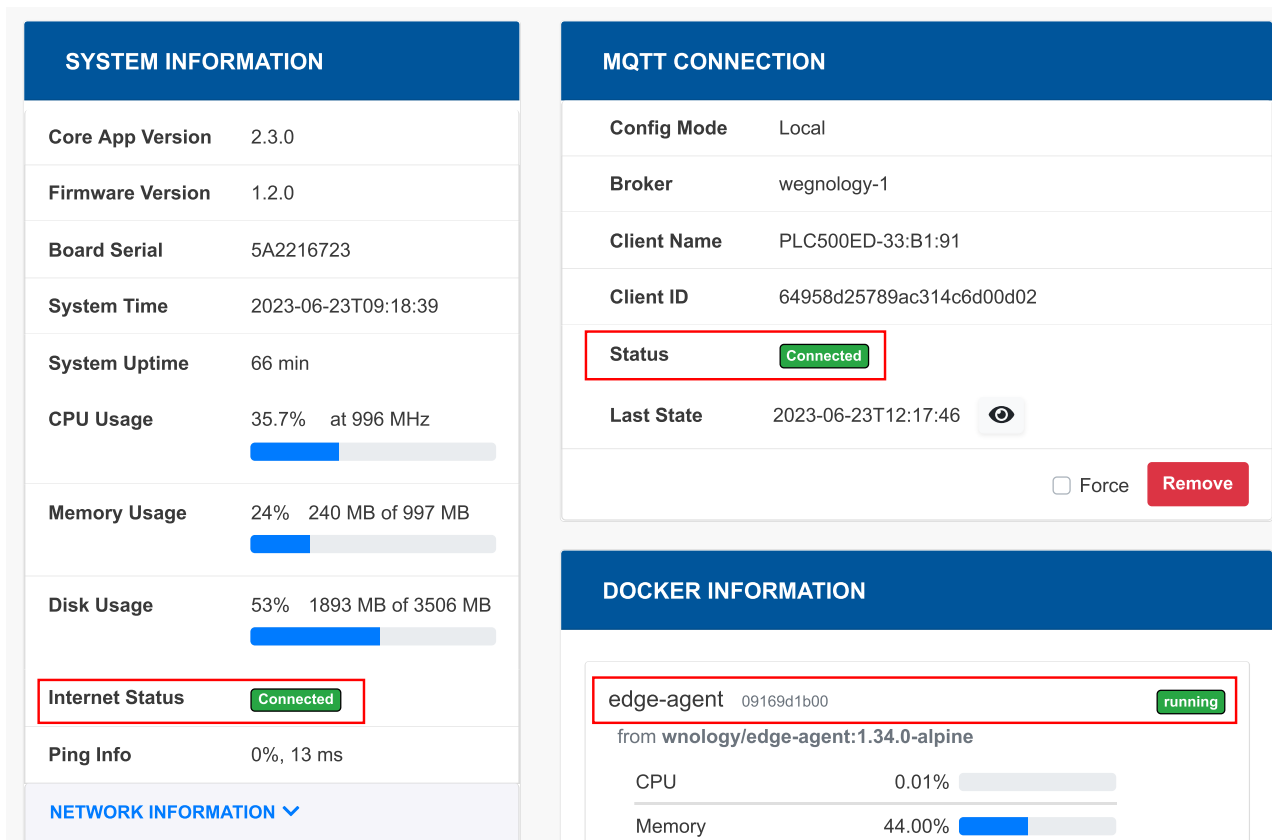


Figura 5.3: Dispositivo conectado a Internet, con una aplicación IoT configurada y una imagen de Docker activa.

5.5 CREACIÓN DE APLICACIONES

Con el producto conectado a Internet y una aplicación IoT configurada, el siguiente paso es crear la aplicación. El PLC500ED permite varios tipos de aplicaciones, tales como:

- **Application Workflows:** estos Workflows se ejecutan en la nube de WEGnology y tienen fines generales de procesamiento y tratamiento de datos. Ofrecen una amplia variedad de triggers para facilitar la adquisición y el manejo de datos de varias fuentes o servicios de datos.
- **Edge Workflows:** Los Edge Workflows se ejecutan en el hardware PLC500ED y son ejecutados por el contenedor de Edge Agent. Proporcionan nodes y triggers adicionales diseñados específicamente para interactuar con periféricos y fuentes de datos locales.
- **Codesys:** Usando la biblioteca **WEGnology** para Codesys, es posible usar el protocolo MQTT para publicar y suscribirse a temas en la plataforma **WEGnology**, así como el contenedor **Edge Agent** según sea necesario. Los bloques de funciones que se comunican directamente con la plataforma tienen el sufijo **WEG**, y los que se usan con Edge Agent tienen el sufijo **Edge**.

Se pueden encontrar ejemplos de aplicaciones en el Capítulo 4, sección 4.5, disponible en los idiomas **Ladder** y **Texto Estructurado (ST)**.



¡NOTA!

Los tres tipos de aplicaciones mencionados se pueden utilizar simultáneamente, teniendo en cuenta las necesidades individuales del usuario, lo que permite aplicaciones versátiles y personalizadas.



WEG Drives & Controls - Automación LTDA.
Jaraguá do Sul - SC - Brasil
Teléfono 55 (47) 3276-4000 - Fax 55 (47) 3276-4020
São Paulo - SP - Brasil
Teléfono 55 (11) 5053-2300 - Fax 55 (11) 5052-4212
automacao@weg.net
www.weg.net