CAREL – Confidential



**REQUIREMENTS SPECIFICATION**

CloudGateMini

Step 2

rev. See history

Index

[Index 2](#_Toc59025227)

[1 Revision 3](#_Toc59025228)

[2 Introduction 4](#_Toc59025229)

[2.1 Scope of the document 4](#_Toc59025230)

[2.2 Definitions, acronyms and abbreviations 4](#_Toc59025231)

[3 Upload of log data read from the connected device 5](#_Toc59025232)

[3.1 Preface 5](#_Toc59025233)

[3.2 MQTT command to start the transfer 5](#_Toc59025234)

[3.3 Transfer of the log 5](#_Toc59025235)

[Background transfer 6](#_Toc59025236)

[3.4 Log contents 6](#_Toc59025237)

[4 Useful links 9](#_Toc59025238)

1. Revision

|  |  |  |  |
| --- | --- | --- | --- |
| Rev. | Rev. date | Author | Note |
| *0.1* | xx/12/2020 | A.Bilato | 1st draft |
| 0.2 | 15/12/2020 | A.Bilato | Added useful link |
| 0.3 | 16/12/2020 | A.Bilato | Added log contents |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Introduction
   1. Scope of the document

This document refers to the realization of some enhancement to the current FW of the CloudGateMini.

* 1. Definitions, acronyms and abbreviations

AP = WiFi access point to connect to

CGM\_AP = gateway that act as an access point  
CGM\_WIFI = CloudGateMini WiFi

CGM = CloudGateMini  
GSM = refer to 2G/4G/NB IoT connection  
FW = firmware

IoT = Internet of Things

MonDev = the device connected to the GME through the RS485 interface

OTA = Over The Air

SW = software  
MFT = Modbus File Transfer

“model file” = a binary file that contains the data needed by the GME to read and write via ModBus the connected device.

MODEL\_TABLE = table defining the meaning of Modbus registers build from “model file”

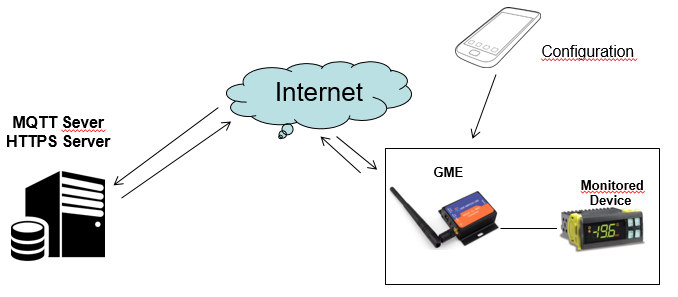
GTW000MWT0 = the GME-WiFi model

GTW000MGP0 = the GME-2G model with CAREL SIM installed

GTW000MGT0 = the GME-2G model without SIM

GTW000M2W0 = the GME-2G+WiFi model without SIM

1. Upload of log data read from   
   the connected device

  
Fig.1

* 1. Preface

Some of the CAREL controllers are able to store some logs and transfer it to a supervisory system through the Modbus File Transfer (MFT) protocol.

To do that with a CGM some modification are requested and are listed below, some must be implemented also server side.  
The block diagram of the system is explained on Fig.1.

* 1. MQTT command to start the transfer

To trigger the transfer of the log the MQTT server send the command

[payload-upload\_logs-req.cbor](payload_examples/payload-upload_logs-req.cbor)

to CGM, the CGM respond …

da stabilire se la conferma dell’avvenuto trasferimento la si vuole alla fine, e ci potrebbero volere ore o se la conferma è all’avvenuta ricezione del commando.

* 1. Transfer of the log

Due metodi possibili

* Chunked transfer encoding, ma ci tocca fare tutto di legno
* Questo sotto, di legno lo stesso, ma il vantaggio è che se qualcosa va storto questo alla fine ci arriva sempre, con calma

The CGM send the file on chunk basis, this due to the limited HW resource.  
The “file” is read from the monitored device block by block, the block size is 200 byte,  
then send the content to the HTTPS server through a POST command.  
The below method is inspired by the HTTP range command in a reverse way.

The POST contains the following fields:   
- “Range: xxxx - yyyy ”;   
a couple of numbers that indicate the start byte and the end byte position on the file.  
Typically you will find 0-199 200-299 etc. the last block contains obviously contains the rest.  
Da capire se serve perchè molto dipende da cosa fa lo script lato server, infatti la demo che ho prodotto fa append e quindi tecnicamente non serve.  
Ma visto il meccanismo con il quale funziona il log mi sa che serve e che lo script debba fare insert del pz al posto giusto (es. header da rileggere alla fine e rimandare).

* “Logdata: xxyy..zz”

The payload, in HEX where xx is the first byte, yy the second and so on.

The reason behind this choice is related to the fact that this data representation is more compact for the CGM.  
Example to transmit the value 15d in HEX is 0F in binary coded for the POST %15  
so a 1 byte extra.  
This method have anyway a little overhead server side where there are two possible strategy, append to the file the HEX value and convert the whole file at the end or convert it chunk on the file when received.  
This 2nd method is preferable because is possible for the cloud to read the header of the file, the 1st chunk and understand the size of the file at the beginning of the transfer, this to understand if the file was fully received or not.

The above method have a quite large overhead because each chunk require to open a connection send the data and close it.

It’s not possible to open and maintain open the connection for the same reason that we have discovered during the transfer of a big file from cloud to the CGM.

Background transfer

The background transfer is theoretically possible, but this feature require defining a new way for the sample of the alarms.  
Currently the alarms are sampled best-effort this means that the Modbus data flow is continuous without any interruption, so that, this method don’t permit to insert a new message to retrieve a data chunk on the middle.  
If we introduce the get of a log chunk the alarm sample rate will decrease, how much ?  
depends on applied strategy.  
Anyway, the total transfer time of the log in background increase a lot compared to the exclusive transfer.

* 1. Log contents

The log are transferred as is without take into account the means of the transferred bytes.  
The Compression header is SENT/NOT SENT(scelta vostra la nostra idea è di mandarlo è parte del file e potrebbe servire lato cloud) as part of the file.

The CGM does not check the file index sent from the cloud, this means that is possible to read virtually any file index, the only check that the CGM perform is on the format and consistency of the Compressor Header [[1]](#UL_1).  
Just for reference the file index currently in use are the ones you will find on [[2].](#UL_2)

1. Useful links

[1]Log File Format

<https://docs.google.com/document/d/19gSBbhbCRT1G5XsI6V0dhK3mYm31KwSa/edit>

Log format Marzolla vecchia versione non aggiornato

[https://docs.google.com/spreadsheets/d/1f\_rMX1lBmSu6YCM6COguO6nWo2AgXItDHF2cLJrEVGw/edit#gid=759605612](https://docs.google.com/spreadsheets/d/1f_rMX1lBmSu6YCM6COguO6nWo2AgXItDHF2cLJrEVGw/edit%23gid=759605612)

[2] Strategia Integrazione

<https://drive.google.com/file/d/1CrJS0_9Qj3wN0NtKzLcX3lbx9s_iJSVm/view?usp=sharing>

---------------

Mail di riferimento

Re: Header per file usando MFT

Alessandro: come detto, trovi la size del payload, quindi dovremmo esserci. Devi considerare solo i campi version e crc dell'header, il resto lo puoi ignorare.

Andrea Cattani: lato OS potremmo inserire questo header prima di quello che già scriviamo al file 3000 (storico allarmi) e 1000 (log).

STc

[https://docs.google.com/document/d/1DoFhnghlsaDkonmvz3HgNPmnK6CfHsbW/edit#](https://docs.google.com/document/d/1DoFhnghlsaDkonmvz3HgNPmnK6CfHsbW/edit)