Energy performance monitoring with Carlo Gavazzi UWP 3.0 and Microsoft Azure IoT

Release 2.1

CARLO GAVAZZI CONTROLS September 4th 2018

Energy performance monitoring with Carlo Gavazzi UWP 3.0 and Microsoft Azure IoT

List of Contents

Abstract
Who should read this document
1. WHY SETTING UP AN IOT BASED SYSTEM BASED ON CARLO GAVAZZI SOLUTIONS FOR MONITORING OF ENERGY
PERFORMANCES
Abstract
LOCAL SOLUTION: UWP 3.0
IOT SOLUTION: THE 4-TIER ARCHITECTURE
UWP 3.0 and Microsoft Azure IoT
2. HOW-TO SET UP AN IOT BASED SYSTEM BASED ON CARLO GAVAZZI SOLUTIONS FOR MONITORING ENERGY
PERFORMANCES
The purpose of this "How to"
AND THE NECESSARY STEPS
FROM UWP 3.0 TO AN SQL DATABASE IN 4 STEPS
1. CREATING AN IOT HUB INTO MICROSOFT AZURE
2. CREATING AN SQL SERVER
3. CREATING A STREAM ANALYTICS PROCESS
4. CROSS TRANSFORMING THE TABLE BY USING THE PIVOT FUNCTION
CONNECTING POWERBI AND CREATING THE DASHBOARD IN 3 STEPS
1. OPENING POWERBI AND CONNECTING TO THE DATA SOURCE
2. CREATING THE DASHBOARD
3. PUBLISHING THE DASHBOARD
DASHBOARD IN ACTION: A REAL-WORLD EXAMPLE
DISCLAIMER

Abstract

Systems Integrators, Energy Managers and Energy Service Companies can greatly benefit from CARLO GAVAZZI's UWP 3.0 datalogger, gateway and web-server. Nowadays the IoT paradigm delivers unlimited potential to convert raw data into valuable information. We're doing our part to make your job easier, quicker and more efficient, while providing you the unbridled power of the cloud.

Who should read this document

This document is meant for system integrators, product managers and CTOs of those companies which are working in the energy efficiency market and want to provide to their customers a scalable solution for energy performance monitoring.

Moreover, Energy Managers looking for energy monitoring solutions for their company will benefit by reading this document.

The IoT paradigm dramatically extends the capabilities of those Cloud based systems meant to interact with the real world: different data sources may be connected to a central hub and the relevant aggregated data may be analysed by different tools and linked to other data repositories; in other words, it is possible to move the focus from data to information. A monitoring system for multiple distributed installations, centralizing data in a common HUB and redistributing information to many systems can be built by putting together different hardware and software components without developing a solution from scratch.

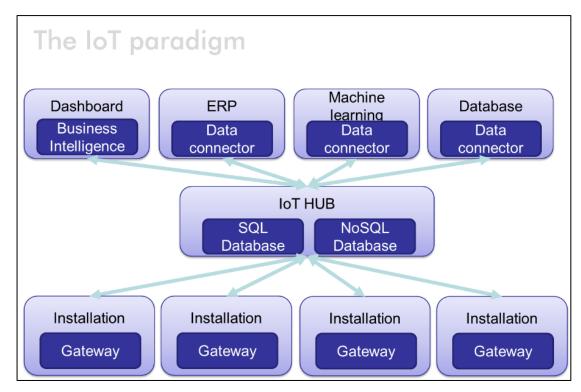


FIGURE 1: THE IOT PARADIGM FOR MONITORING DISTRIBUTED INSTALLATIONS

Basically, this document is split into 2 sections:

- 1. Section-1: explaining the overall solution to decision makers and energy managers
- 2. Section-2: going deeper into the details of How-To creating an online dashboard solution based on UWP 3.0

1. why setting up an IoT based system based on Carlo Gavazzi solutions for monitoring of energy performances.

Abstract

Energy performance monitoring is a matter of:

- 1) Installing and connecting energy meters or energy analysers to the target loads
- 2) Gathering data from meters and storing them locally
- 3) Re-transmitting data to remote systems for aggregating them, creating KPIs for benchmarking and reporting to decision makers and Energy Managers

Local solution: UWP 3.0



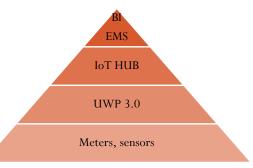
UWP 3.0 is a modular system that records, monitors and transmits analogue and digital signals from an industrial, commercial or residential installation with a specific focus on energy efficiency. The system includes a web server with a powerful and intuitive user interface to monitor data and set up the system. Data can be transmitted using various protocols (FTP, HTTP, Modbus TCP/IP) and via wired or wireless connection.

UWP 3.0 includes everything is needed to monitor a local installation, by gathering data from meters, logging them, displaying them via the web server and sending them remotely. Nonetheless, very often, it is necessary to:

- 1) Centralize data from multiple locations into a common repository
- 2) Integrate measured data with information from other sources
- 3) Analyse data with business intelligence tools

In this case the IoT paradigm provides a solution by connecting together different tiers of a common monitoring pyramid.

IoT solution: the 4-tier architecture



This is a 4-tier architecture, including meters, sensors, edge units (the UWP 3.0) and the Business Intelligence (BI) or Energy Management System (EMS) tiers.

The Edge unit (the UWP 3.0) gather data from the field and delivers them to the IoT HUB. Energy data management solutions developed in the cloud communicate with the IoT HUB and provide users with the needed functions to convert raw data into information.

UWP 3.0 and Microsoft Azure IoT

 Microsoft Azure
 UWP 3.0 is Microsoft Azure Certified for IoT.

 You can find more details about UWP 3.0 certification at the Microsoft link here:

 https://catalog.azureiotsolutions.com/details?title=UWP30RSEXXX&source=home-page

Variables gathered from the connected devices (meters, sensors and others) are converted into JSON format and sent to the Microsoft Azure IoT HUB with the necessary security and reliability.

By having data available on Microsoft Azure, users can leverage the powerful Microsoft tools for:

- a) Integrating other data source data
- b) Sharing information with other systems
- c) Using the best Business Intelligence tools to dig into data

Please find more information about the Azure IoT solutions here: <u>https://azure.microsoft.com/en-gb/suites/iot-suite/</u>

2. How-To set up an IoT based system based on Carlo Gavazzi solutions for monitoring energy performances.

The purpose of this "How to" ...

Integrating data pushed by UWP 3.0 into Microsoft Azure IoT, allows to put in place a monitoring system in a matter of hours without developing any software; by leveraging the powerful tools provided by Microsoft, it is possible to create relationships between UWP 3.0 data and information coming from different sources.

Besides, the use of Microsoft unlimited scalability options allows to match whatever use cases, from single site.

Even though once data are available into Microsoft Azure, there are dozens of different integration paths: the purpose of this document is to show how to analyse them with Microsoft PowerBI.

For further information about PowerBI, please check the official Microsoft PowerBI documentation: https://powerbi.microsoft.com/en-us/

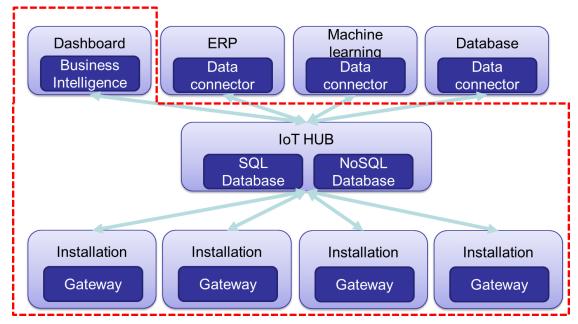


FIGURE 2: OUR CONTEXT, FROM INSTALLATIONS TO BUSINESS INTELLIGENCE

...and the necessary steps

The necessary steps are:

- 1) Log it into a proper Microsoft Azure account
- 2) Activate an IoT Hub from the relevant menu
- 3) Connect the target UWP 3.0 to the IoT HUB
- 4) Create an SQL Server database from the relevant menu
- 5) Create the target table(s) into the DB created at point -4-
- 6) Create a Stream Analytics item from the relevant menu, so as to connect the IoT HUB to the SQL Server
- 7) Use a Dashboard tool like PowerBI to connect to the SQL Server, create and publish the desired dashboards

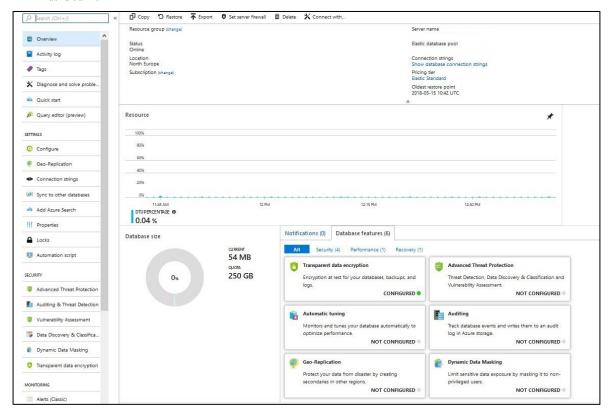


FIGURE 3: MICROSOFT AZURE CONTROL PANEL

We will see the relevant details in the next pages.

From UWP 3.0 to an SQL database in 4 steps

1. Creating an IoT HUB into Microsoft Azure

By using the Resources menu, it is possible to create the IoT HUB; an IoT HUB is necessary to create a single contact point on which the gateways can push data via MQTT; once the IoT HUB has been created, the relevant connection string must be set into the gateway so to create the link and start the communication.

Microsoft Azure		${\cal P}$ Search resources, services, and docs	× 🗘 >_ 🏟 😳 Ø	hosting.box@gavazzia DIRECTORY PREDEFINITA
« + Create a resource	CGC-IoT-HUB - Properties			* ×
i≡ All services	P Search (Ctr(+/) «	NAME		^
+ FAVORITES	X Overview	CGC-IoT-HUB		
🔚 Dashboard	Activity log	RESOURCE TYPE Microsoft.Devices/lotHubs		
All resources	Access control (IAM)	Wild osoft. Devices/formubs		
📦 Resource groups	🛷 Tags	LOCATION		
🌲 Quickstart Center	🗲 Events	West Europe		
🔇 App Services	SETTINGS	LOCATION ID westeurope		
Function Apps	📍 Shared access policies			
🧧 SQL databases	O Pricing and scale	RESOURCE ID /subscriptions/		lubs/CGC-loT-HUB
🬌 Azure Cosmos DB	Operations monitoring	Provide a second		
Virtual machines	📴 IP Filter	RESOURCE GROUP		
Load balancers	🟓 Certificates	CGC-Resource-Group		
Storage accounts	E Properties	RESOURCE GROUP ID		
	Locks	/subsci	-Group	
• Virtual networks	🖳 Automation script	SUBSCRIPTION		
Azure Active Directory	EXPLORERS	Versione di valutazione gratuita		
Monitor		SUBSCRIPTION ID		
🜩 Advisor	Query explorer	/subscription 3d3		

FIGURE 4: IOT HUB PARAMETERS AFTER THE SET-UP

For more information about the JSON format of data that UWP 3.0 pushes to internet, please check the following document: <u>http://www.productselection.net/PDF/UK/UWP3.0_IoT_Protocols_R1.pdf</u>

When data are available within Azure IoT Hub as JSON containers, the system integrators can move them to an SQL or an SQL database and use the available tools of MS Azure to display, convert, aggregate and analyse them according to the needs.

2. Creating an SQL Server

An SQL server is necessary to process historical data as a foundation for dashboards; IoT HUB data are unstructured (they have been delivered as JSON files) and can hardly be used for creating efficient queries. For these reasons, it is necessary to:

- a) Create an SQL Server, by using the specific menu item in the Microsoft Azure control panel; the set-up depends on the available credit, and the application needs;
- b) Create a table in the SQL Server, on which the data stream will be targeted;
- c) Create a Pivot View, linked to the data table, meant to define a columns-based data structure which can be efficiently used by the dashboards.

The table may be created with a query like this one:

create table UWP30 (
varDatetime datetime,
GatewaySN char(50),
DeviceName char(50),
varName char(50),
varType char(50),
varEngUnit char(50),
varValue real
);

Microsoft Azure		ρ Search resources, services, a	and do	<i>ct</i>	×	L₽ >_	\$ © 0	R t	osting.box@gavaz	zia
witcrosoft Azure	Home > SOL databases > CGC-SOL-SERVER	- Search resources, services, a	<i>ina</i> uo	••	^	~~ ~_		ΨY	DIRECTORY PREDEF	
+ Create a resource	SQL databases « * × Directory predefinita	GGC-SQL-SERVER								* ×
i∃ All services	➡ Add III Edit columns ···· More			🗗 Сору	්ට Restore	⊼ Export	O Set server firewall	菌 Delete	🗙 Connect with	
- * FAVORITES	Filter by name	Cverview	•	Resource gr CGC-Resour Status			Server na cgc-si Elastic di	ime itabase pool	iws.net	^
🛄 Dashboard	NAME 🔍	Activity log		Online			No elasti			
All resources	GGC-SQL-SERVER	🖉 Tags		France Cent Subscription			Show da Pricing t	tabase connec er	tion strings	
📦 Resource groups		X Diagnose and solve problems		Subscription	ID		Oldest re	S0: 10 DTUs store point		
🌲 Quickstart Center		📣 Quick start				98d3	2018-05-	21 07:31 UTC		
S App Services		Duery editor (preview)		Resource						*
Function Apps	SETTINGS		100%							
👼 SQL databases		O Configure		80%						
🬌 Azure Cosmos DB		🖤 Geo-Replication		40%						
Virtual machines		 Connection strings 		20%						
🚸 Load balancers		Sync to other databases		0%						
Storage accounts		🚳 Add Azure Search		DTU PERC		1:15 PM	1:30 PM	1:4	5 PM	2 PM
··· Virtual networks		Properties		0 %						
Azure Active Directory		Locks		Database siz	e					
 Monitor 		Automation script					5 MB			
		SECURITY					QUOTA 100 GB			
Advisor		v		¢	0%		100 00			>

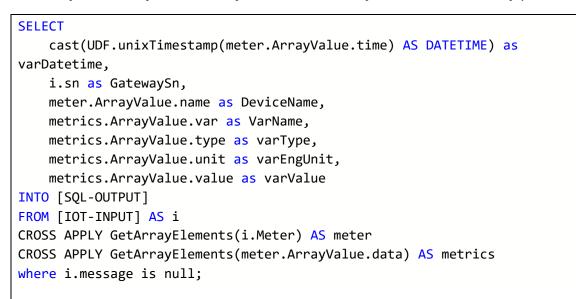
Figure 5:The SQL Server has been correctly configured

3. Creating a Stream Analytics process

By using the Resources menu in Microsoft Azure Control Panel, a Streaming Analytics must be created to join:

- a) INPUT: the IoT HUB
- b) OUTPUT: the SQL Server's table

The link is possible via a specific function (please find here an example based on the UWP 3.0 payload):



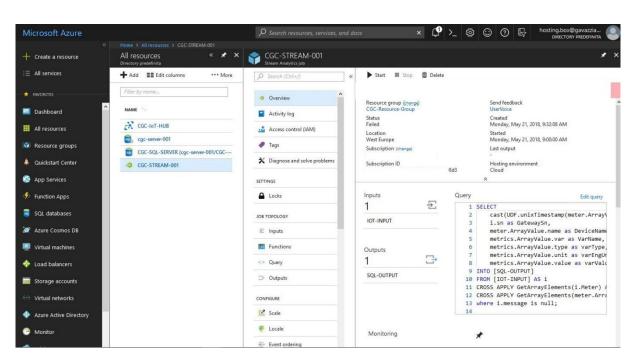


FIGURE 6: THE STREAM ANALYTICS CONFIGURATION PAGE

4. Cross transforming the table by using the PIVOT function

Usually it is necessary to cross-transform the Database table in which values are stored because a columns-based structure is more efficient than a row-based layout when dashboards are the targets.

In our case, the target is to extract energy data and transform the variable names (kWh, kvarh types) into columns.

This is the relevant query which can be implemented as a VIEW into the SQL Server database or used as the datasource query into the dashboard tool (i.e. Power BI, see later on).

```
SELECT *
FROM (SELECT
                    varDateTime,
                    GatewaySN,
                    DeviceName,
                    varName,
                    vartype,
                    varValue
        FROM
               UWP30
          where varType =0
         ) as SourceTable
PIVOT
(
   Avg(varValue)
   FOR varName in
   ([kWhac],[kWhacn],[kWhl1],[kWhl2],[kWhl3],[kvarh],[kvarhn])
) AS AvgVarValue
ORDER BY varDateTime
```

Connecting PowerBI and creating the dashboard in 3 steps

PowerBI is a powerful dashboarding tool from Microsoft which allows user to build dashboards by connecting to a wide range of data sources, analyse them locally, sharing and/or publishing them on the web.

For further information please check here: <u>https://powerbi.microsoft.com/en-us/desktop/</u>

1. Opening PowerBI and connecting to the data source

By selecting NEW SOURCE, it is possible to set the connection parameters to the AZURE SQL SERVER and set the connection query implementing the PIVOT function (see the relevant point above).

By selecting IMPORT, it is possible to update data on demand, or set a scheduler; this is the less resource consuming option (pay care to this part because querying and downloading data has a cost).

File Home Transform Add Column View He	.p				^ (
Close & New Recent Enter Apply* Source* Sources* Data	Refresh Preview + Manage + Columns + Co		Data Type: Date/Time * Data Type: Date/Time *	Generation → Generation → Generation → Generation → Generation → Generation →	
Close New Query Data Sources Parameters			×	Combine	
Queries [1] < . varDateTime < A ⁸ C	SQL Server database			QUERY SETTINGS	
	Server ③			^	
	cgc-s dows.net			PROPERTIES	
	Database			Name	
4 13/05/2018 02:45:00 BN1	CGC			Query1	
5 13/05/2018 02:45:00 BN1				<u>All Properties</u>	
6 13/05/2018 02:45:00 BN1	Data Connectivity mode 🛈			▲ APPLIED STEPS	
7 13/05/2018 02:45:00 BN1	Import DirectQuery				
8 13/05/2018 02:45:00 BN1	O Directodery			Source	*
9 13/05/2018 02:45:00 BN1	Advanced options				
10 13/05/2018 02:45:00 BN1	Command timeout in minutes (optional)				
11 13/05/2018 02:45:00 BN1					
12 13/05/2018 02:45:00 BN1	SQL statement (optional, requires databas	se)			
13 13/05/2018 02:45:00 BN1	vartyp	e,			
14. 13/05/2018 02:45:00 BN1	FROM VMUC	lue	^		
15 13/05/2018 02:45:00 BN1	where varType =	0			
16 13/05/2018 02:45:00 BN1) as SourceTable				
17 13/05/2018 03:00:00 BN1	PIVOT				
18 13/05/2018 03:00:00 BN1	(
19 13/05/2018 03:00:00 BN1	Avg(varValue) FOR varName in				
20 13/05/2018 03:00:00 BN1		hl2],[kWhl3],[kvarh],[kvarhn])			
21 13/05/2018 03:00:00 BN1) AS AvgVarValue ORDER BY varDateTime		~		
22 13/05/2018 03:00:00 BN1					
23 13/05/2018 03:00:00 BN1	Include relationship columns				
24 13/05/2018 03:00:00 BN1	Navigate using full hierarchy				
25 13/05/2018 03:00:00 BN1	Enable SQL Server Failover support				
26 13/05/2018 03:00:00 BN1					
27 13/05/2018 03:00:00 BN1			OK Cancel		
28 13/05/2018 03:00:00 BN1			Carter		
29 13/05/2018 03:00:00 BN1					
30 13/05/2018 03:00:00 BN17	/00015001H	Production_line_F_EM330	0		
31 13/05/2018 03:00:00 BN12	00015001H	Production_line_C_EM111	0		
32 13/05/2018 03:00:00 BN17	700015001H	Main_outside_WM30	0	~	
33 13/05/2018 03-15-00 BN17	/00015001H	Production line A FM111	0		
				>	

FIGURE 7: THE DATA SOURCE CONNECTION

2. Creating the Dashboard

By using the tools of PowerBI, it is possible to create dashboards with filters and sliders and, by selecting the preferred charts, it is also possible to join different datasources.

The description of PowerBI functions is beyond the scope of this document; the relevant documentation is available here: https://powerbi.microsoft.com/en-us/desktop/

For advanced users, PowerBI provides DAX embedded language: <u>https://docs.microsoft.com/en-us/power-bi/guided-learning/introductiontodax?tutorial-step=1</u>

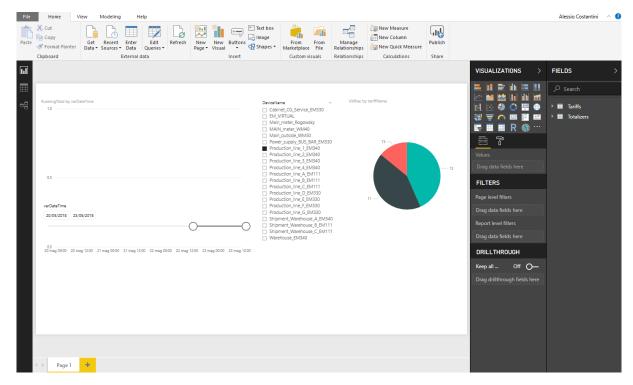


FIGURE 8: DASHBOARD EDITING

3. Publishing the Dashboard

By selecting PUBLISH, the user can publish the dashboard on the selected workspace.

It is necessary to subscribe to PowerBI to publish dashboards. Afterwards the project is uploaded to the selected PowerBI account and then it can be:

- a) Used by the account owner
- b) Shared with other PowerBI accounts
- c) Published on a website

The full description of PowerBI web capabilities is beyond the scope of this document; please check here for further information: <u>https://powerbi.microsoft.com/en-us/desktop/</u>

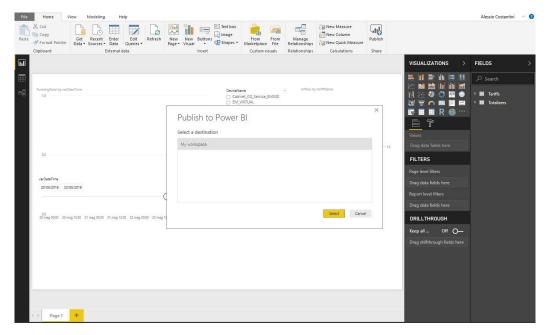


FIGURE 9: PUBLISHING A DASHBOARD

Dashboard in action: a real-world example

By opening the following link, it is possible to see a dashboard connected to a real installation powered by UWP 3.0 and using the above tools to publish the relevant data:

https://app.powerbi.com/view?r=eyJrIjoiNzM2N2FiNTQtNzBiZS00NTM5LWE2ZGEtYWQwMWE3MGI4M DM4IiwidCI6IjkwMjNiZWM1LWZiYjMtNDUzMi1iNzMwLWFhZWI3NmQ1OTBjYSIsImMiOjl9

Disclaimer

Microsoft, PowerBI, Azure, SQL Server are registered trademarks of Microsoft. Carlo Gavazzi has no commercial relationship with Microsoft. Therefore all the Microsoft solutions mentioned in the document must be purchased and supported through the Microsoft commercial channels. CARLO GAVAZZI MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, AS TO THE INFORMATION ON THIS DOCUMENT. This CONTENT is provided "as-is." Information and views expressed in this document, including URLs and other Internet website references, may change without notice. This document does not provide you with any legal rights to any intellectual property in any Carlo Gavazzi or Microsoft product. You may copy and use this document for your internal, reference purposes only.