Alleantia
ALL <THINGS> JOINED

MANUALE D'USO
PROTOCOLLO MQTT


Alleantia
ALL <THINGS> JOINED

Indice

Introduzione

1. Configurazione Impianto

Il dispositivo gateway pubblica i dati dei devices monitorati sul broker MQTT del cliente.

Una volta configurato il software Alleantia in campo ed inseriti gli indirizzamenti del broker e la frequenza di invio dati, il sistema in automatico provvederà all'invio degli stessi.

Preventivamente occorrerà anche selezionare le variabili, allarmi ed eventi di interesse da inoltrare tramite il servizio MQTT.

2. Configurazione e mappatura iniziale

Inizialmente occorre fare la scansione della configurazione e mappatura dell'impianto in modo da ricostruirsi lo stato dello stesso ed in particolare:

- il n. di dispositivi gateway presenti in campo: ognuno avrà il proprio Serial Id distintivo.

Per ognuno dei gateway presenti in campo occorrerà una prima fase di interrogazioni per creare una lookup table dove conservare le chiavi di decodifica dei dati.

Tali interrogazioni sono riportate nel capitolo 3 paragrafo 3.2 in poi della documentazione tecnica del manuale MQTT e consentono di comprendere e mappare:

- il numero dei devices presenti in campo e loro nomenclatura univoca (dev Id): trattasi del n° di apparecchiature/macchine monitorate dalla specifica licenza sw;
- il numero di variabili che sottostanno ad ogni device (var id): tali variabili sono i dati di interesse quali velocità, temperature, etc trasmesse dalla macchina.

Una volta ricostruita la mappatura dell'impianto mancherà solo la ricezione del valore della variabile (value) che viene comunicato autonomamente in automatico dal dispositivo nel **Topic Telemetria**.

3. Topics

I dati pubblicati dal sistema sono suddivisi in *Topics*.
Ogni gateway pubblicherà messaggi su 3 Topics diversi:

- Topic Allarmi
- Topic Eventi
- Topic Telemetria

La configurazione iniziale invece prevede un quarto Topic su cui il gateway rimane in ascolto e attende le richieste dall'esterno secondo la sintassi prevista (cap. 3 del manuale):

- Topic Commands.

Ogni Topic si presenta con il seguente pattern: *Serial Id/Telemetry, Serial Id/Alarm, Serial Id/Events, Serial Id/Commands.*

Quindi, il Topic Commands verrà usato al fine di realizzare le richieste iniziali da parte dell'utente utili per ricostruire il lookup dell'impianto.

Introduzione

Tutte le risposte da parte del gateway avverranno sui Topic di pubblicazione sopra citati ed in particolare il Topic Telemetry.
Solo le informazioni relative ad allarmi o eventi impostati a sistema (software Alleantia) verranno pubblicati sui relativi **Topic Alarms** e **Events**.

Nota:
Ogni aggiunta in campo di dispositivi monitorati e variazione della configurazione implica l'aggiornamento del lookup e la replica delle richieste di informazioni al **Topic Commands**.

Sintesi finale

Occorre inizialmente fare delle chiamate ai gateway in campo al fine di stabilire quali siano le macchine monitorate e quali variabili ogni macchina comunica.

A seguito dell'avvenuta mappatura dell'impianto si potrà andare a leggere i valori delle variabili di cui si conoscerà origine (macchina e gateway).

E previsto quindi un set di chiamate iniziali sul **Topic Commands** ed il gateway risponderà sul **Topic Telemetry**.

Seguiranno gli invii ricorrenti delle informazioni automatizzate sui 3 **Topics** di pubblicazione.

1 Descrizione Servizio MQTT

L'IoT Scada fornisce il servizio di inoltro messaggi al cloud mediante il protocollo MQTT.
Se la licenza in possesso è abilitata e se il servizio è stato attivato, l'IoT Scada invierà periodicamente al broker MQTT configurato dei messaggi di telemetria sotto forma di stringa in formato JSON.

L'IoT Scada fornisce un meccanismo di configurazione dei permessi (lettura/scrittura/inoltro allarmi) relativi alle variabili da inoltrare, pertanto verranno inoltrate tramite i messaggi di telemetria solamente le variabili per cui è stato preventivamente fornito tale permesso. Anche l'operazione di scrittura (se logicamente possibile) sarà effettuata solamente se è stato preventivamente abilitato tale permesso nella configurazione.
Esistono attualmente tre tipologie di messaggi inoltrabili in maniera automatica dal dispositivo, che differiscono per quantità di informazioni fornite e per sintassi utilizzata (e quindi per dimensioni del messaggio).

Nell'IoT Scada ogni singola variabile raccolta dal campo è identificata univocamente dalla coppia devId (identificativo del device di appartenenza della variabile) e varId (identificativo della variabile). È possibile che alcune variabili abbiano un devId non valorizzato: queste sono le cosiddette variabili d'Impianto (chiamate Misure Personalizzate nell'IoT Scada) che non possiedono tale identificativo per la natura stessa per cui sono state create: esse non appartengono infatti ad alcun dispositivo, ma sono il risultato di un'elaborazione fatta su più variabili, tendenzialmente provenienti da dispositivi diversi (esempio: la somma dei kWh letti da N inverter collegati all'IoT Scada).

Il principio di funzionamento per far si che venga inoltrata solamente la minima quantità di dati prevede che sia effettuata in fase di inizializzazione una prima ed unica interrogazione inoltrando uno o più messaggi in formato JSON (esplicato in seguito in questo manuale) verso il broker per ottenere in risposta la configurazione del dispositivo, device connessi, configurazione di variabili, allarmi, eventi, e con questi costruirsi una lookup table.
In questo modo l'utilizzatore avrà a disposizione le chiavi per identificare univocamente i dati inoltrati automaticamente nei successivi messaggi di telemetria cosicché tali informazioni non debbano essere inoltrate nuovamente ad ogni messaggio.

2 Messaggi di telemetria

I messaggi di telemetria inoltrati automaticamente a frequenza impostabile nell'interfaccia del dispositivo hanno la seguente sintassi:

2.1 Messaggio completo

```
{
  "telemetryDataList":[
    {
      "date":"Jan 8, 2018 10:50:30 AM", //java.util.Date
      "category":"main", //String
      "blockId":"B2", //String
      "description":"Ingresso digitale 3, overflow conteggio parziale",
      "deviceDescription":"IoT server-11x",
      "dataType":"BOOLEAN", //String
      "devId":3, //int
      "varId":171, //int
      "value":false, //Object(String,Boolean,Numeric)
      "quality":true  //boolean
    }, //..... N variabili
  ],
  "devSn": "IOTSPIXXXXXXXXX",  //String
  "onTime": " Jan 8, 2018 10:50:34 AM"  //java.util.Date
}
```

2.2 Messaggio normale

```
{
  "telemetryDataList":[
    {
      "date":"Jan 8, 2018 11:13:27 AM",
      "devId":3,
      "varId":171,
      "value":false,
      "quality":true
    }, //..... N variabili
  ],
  "devSn": "IOTSPIXXXXXXXXX",
  "onTime": " Jan 8, 2018 11:13:27 AM "
}
```

2 Messaggi di telemetria

La lista di telemetryData restituita dall'IoT Scada è un JSON composto da:

• devId – identificatore del device a cui appartiene la variabile (può essere null in caso di variabili d'impianto)
• varId – identificatore della variabile
• value – contiene il valore letto per la variabile
• quality – valore booleano che indica se l'ultimo tentativo di lettura ha avuto successo o meno, quindi se il valore contenuto in value è attuale o potenzialmente datato
• date – data a cui risale l'ultima lettura corretta della variabile

Ogni messaggio avrà poi due campi di intestazione:

• devSn – contiene il seriale univoco del dispositivo che sta inoltrando le variabili
• onTime – data di inoltro del messaggio

N.B. È possibile discriminare tra queste due tipologie di messaggi (messaggio completo e messaggio normale) semplicemente spuntando o meno nell'interfaccia dell'IoT Scada la voce **"Il messaggio inoltrato conterrà solamente le informazioni essenziali"**.

2.3 Json Time series (JTS)

```
{
  "samples":5,
  "samplesDelay":0,
  "startTime":1515406966223,
  "endTime":1515406970224,
  "telemetryDataList":[
    {
      "devId":3,
      "varId":171,
      "values":[false, false, true, true, false],
      "quality":[0,1,1,1,1]
    }, //..... N variabili
  ],
  "devSn": "IOTSPIXXXXXXXXX",
  "onTime": " Jan 8, 2018 11:13:27 AM "
}
```

Il messaggio si apre con i seguenti parametri di intestazione:

• samples (s) – numero di campionamenti effettuati nell'unità di tempo (settabili dall'interfaccia Servizio MQTT);
• samplesDelay – Intervallo di tempo del singolo campionamento (ricavato secondo la formula **(sT-eT)/s**);
• startTime (sT) – tempo di inizio della serie di campionamenti;

2 Messaggi di telemetria

• endTime (eT) – timestamp finale della serie di campionamenti.
La lista di telemetryData restituita dall'IoT Scada è un JSON composto da:

• devId – identificatore del device a cui appartiene la variabile (può essere null in caso di variabili d'impianto);
• varId – identificatore della variabile;
• value – contiene il valore letto per la variabile;
• quality – valore booleano che indica se l'ultimo tentativo di lettura ha avuto successo o meno, quindi se il valore contenuto in value è attuale o potenzialmente datato;
• date – data a cui risale l'ultima lettura corretta della variabile.

Ogni messaggio avrà poi due campi di intestazione:
• devSn – contiene il seriale univoco del dispositivo che sta inoltrando le variabili;
• onTime – data di inoltro del messaggio.

N.B. È possibile scegliere la seguente tipologia di messaggio semplicemente spuntando nell'interfaccia dell'IoT Scada la voce *"JTS (Json Time Series)"*. Una volta effettuata la scelta è possibile impostare il sumero di campionamenti effettuati nell'intervallo di tempo (Samples).

3 Interrogazioni all'IoT Scada

Come precedentemente accennato, l'IoT Scada può rimanere in ascolto in attesa di messaggi pubblicati sul broker e rispondere puntualmente a richieste di informazioni qui effettuate.

Il dispositivo IoT Scada rimarrà in ascolto sul broker *devSn/commands* e risponderà alle richieste sul broker *devSn/telemetry*.

Tutti i messaggi riconosciuti dall'IoT Scada condividono una stessa struttura di base, da compilare diversamente a seconda del tipo di richiesta.

Il messaggio riconosciuto dall'IoT Scada possiede la seguente struttura:

```
{
   "component":"",
   "operation":"",
   "devId": [],
   "varId": [],
   "startTime":"",
   "endTime":"",
   "value":""
}
```

Ovviamente, come verrà spiegato in seguito, alcuni campi dovranno essere compilati per determinate chiamate, e lasciati vuoti in altre, pena il mancato riconoscimento di tale messaggio e quindi di risposta.

La struttura di base è composta da:

• component – (obbligatorio) - enum dei componenti coinvolti (INFO, DEVICES, ALARMS, EVENTS);
• operation – (obbligatorio tranne per component INFO) - enum delle operazioni possibili (LIST, DATA, CONFIG, LOGDATA, SET, ACTIVE, HISTORY);
• devId – (opzionale) – lista di deviceIds;
• varId – (opzionale) – lista delle variabili;
• startTime - (opzionale) – estremo iniziale dell'intervallo temporale su cui estrarre i dati;
• endTime - (opzionale) – estremo finale dell'intervallo temporale su cui estrarre i dati;
• value - (opzionale) – valore da scrivere nella variabile.

3.1 Informazioni del sistema

3.1.1 Informazioni sull'IoT Scada

Per ottenere tali informazioni il messaggio JSON da inoltrare è:

```
{
   "component":"INFO"
}
```

3 Interrogazioni all'IoT Scada

L'IoT Scada risponderà con una serie di informazioni relative all'IoT Scada:

```
{
  "uuid": "49ab91753-9b94-3d2e-8ff6-17d3d5113606",
  "hwModel": "4.0.5-SNAPSHOT",
  "name": "IoT Gateway",
  "webAppVersion": "4.0.3",
  "devSn": " IOTSPIXXXXXXXXX ",
  "onTime": "Sep 20, 2017 5:23:12 PM"
}
```

3.2 Configurazione dei devices collegati

Per ottenere tali informazioni il messaggio JSON da inoltrare è:

```
{
   "component":"DEVICES",
   "operation":"LIST",
   "devId": [ 2, 3, 4 ]    //OPZIONALE
}
```

Il parametro lista devId è facoltativo e serve per filtrare le informazioni: omettendo tale parametro il dispositivo fornirà le informazioni di tutti i device. Impostandolo, verranno restituite solamente le informazioni di quelli selezionati.

L'IoT Scada risponderà con una serie di informazioni relative al device:

```
{
  "devices": [
    {
      "devId": 2,
      "description": "Fanuc 750",
      "linked": true
    },
    {
      "devId": 3,
      "description": "Fanuc 545",
      "linked": true
    },
    {
      "devId": 4,
      "description": "MCM-cnc",
      "linked": false
    }
  ],
  "devSn": " IOTSPIXXXXXXXXX ",
  "onTime": "Sep 20, 2017 5:23:12 PM"
}
```

4 Informazioni sulle variabili

4.1 Configurazione delle variabili

Per ottenere tali informazioni il messaggio JSON da inoltrare è:

```
{
   “component”:”DEVICES”,
   “operation”:”CONFIG”,
   “devId”: [ 63 ],    //OPZIONALE
   “varId”: [ 40, 41, 39 ]    //OPZIONALE
}
```

Il parametro lista devId è facoltativo e serve per filtrare le informazioni: omettendo tale parametro il dispositivo fornirà le informazioni di tutti i device. Impostandolo, verranno restituite solamente le informazioni di quelli selezionati. Il parametro lista varId è facoltativo. Se specificato, dovrà esserci **un solo** devId specificato nel corrispettivo parametro. Il dispositivo fornirà solamente le informazioni relative ai varId selezionati per tale dispositivo.

L’IoT Scada risponderà con una serie di informazioni relative alle variabili:

```
{
  “varConfigList”: [
    {
      “devId”: 63,
      “varId”: 21,
      “description”: “Actual executed program name”,
      “dataType”: “String”,
      “minimum”: “null”,
      “maximum”: “null”,
      “category”: [  “main”    ],
      “alarmable”: false,
      “writable”: false
    },
    {
      “devId”: 63,
      “varId”: 23,
      “description”: “Automatic mode selection”,
      “dataType”: “Numeric”,
      “minimum”: “null”,
      “maximum”: “null”,
      “category”: [  “main”    ],
      “alarmable”: false,
      “writable”: false
  ],
  “devSn”: “ IOTSPIXXXXXXXXX”,
  “onTime”: “Feb 20, 2017 4:11:13 PM”
}
```

4 Informazioni sulle variabili

4.2 Dati attuali delle variabili

Per ottenere tali informazioni il messaggio JSON da inoltrare è:

```
{
   “component”:”DEVICES”,
   “operation”:”DATA”,
   “devId”: [ 63 ],    //OPZIONALE
   “varId”: [ 23, 430 ]    //OPZIONALE
}
```

Il parametro lista devId è facoltativo e serve per filtrare le informazioni: omettendo tale parametro il dispositivo fornirà le informazioni di tutti i device. Impostandolo, verranno restituite solamente le informazioni di quelli selezionati. Il parametro lista varId è facoltativo. Se specificato, dovrà esserci **un solo** devId specificato nel corrispettivo parametro. Il dispositivo fornirà solamente le informazioni relative ai varId selezionati per tale dispositivo.

L’IoT Scada risponderà con una serie di informazioni relative alle variabili:

```
{
  “variablesList”: [
     {
       “devId”: 63,
       “varId”: 23,
       “value”: “MDI”,
       “decodedValue”: “MDI”,
       “date”: “Sep 20, 2017 5:48:57 PM”,
       “quality”: false
    },
    {
       “devId”: 63,
       “varId”: 430,
       “value”: false,
       “date”: “Sep 20, 2017 5:48:57 PM”,
       “quality”: false
    }
  ],
  “devSn”: “ IOTSPIXXXXXXXXX”,
  “onTime”: “Feb 20, 2017 4:29:46 PM”
}
```

4 Informazioni sulle variabili

4.3 Dati di Log

Per ottenere tali informazioni il messaggio JSON da inoltrare è:

```
{
   “component”:”DEVICES”,
   “operation”:”LOGDATA”,
   “devId”: [ 63 ],     //OBBLIGATORIO
   “varId”: [ 23 ],     //OBBLIGATORIO
   “startTime”: 0,    //OPZIONALE
   “endTime”: 123456789    //OPZIONALE
}
```

I parametri lista devId e varId sono obbligatori. Dovrà esserci un solo valore in ogni lista. I parametri startTime e endTime sono facoltativi. Tali parametri applicano un filtro temporale sui dati da restituire. I valori da inserire sono espressi in millisecondi a partire dal 1/1/1070 0:00 GMT.

L’IoT Scada risponderà con una serie di informazioni relative alle variabili:

```
{
        “devId”: 63,
        “varId”: 23,
        “value”: “MDI”,
        “date”: “Sep 8, 2017 1:37:14 PM”,
        “quality”: true
     },
     {
        “devId”: 63,
        “varId”: 23,
        “value”: “MDI”,
        “date”: “Sep 8, 2017 1:40:14 PM”,
        “quality”: true
     }
  ],
  “devSn”: “ IOTSPIXXXXXXXXX”,
  “onTime”: “Feb 20, 2017 5:02:49 PM”
}
```

I campi devId e varId sono facoltativi. Se lasciati vuoti, l’IoT Scada risponderà con la lista di tutte le variabili; se specificato uno o più devIds, l’output verrà filtrato restituendo solamente quelli richiesti. Se specificati anche una lista di varId, il devId dovrà essere univoco: verranno restituite solamente le variabili con tale devId e tali varId.

4 Informazioni sulle variabili

4.4 Settaggio di una variabile

Per ottenere tali informazioni il messaggio JSON da inoltrare è:

```
{
   “component”:”DEVICES”,
   “operation”:”SET”,
   “devId”: [ 31 ],    //OBBLIGATORIO
   “varId”: [ 47],      //OBBLIGATORIO
   “value”: 10     //OBBLIGATORIO
}
```

I parametri lista devId e varId sono obbligatori. Dovrà esserci un solo valore in ogni lista.
Il parametro value indica il valore a cui si vuole settare la variabile.

L’IoT Scada risponderà con una serie di informazioni relative alle variabili:

```
{
  “accepted”: true,
  “description”: “Accepted”,
  “devSn”: “ IOTSPIXXXXXXXXX”,
  “onTime”: “Feb 20, 2017 5:02:49 PM”
}
```

5 Scambio dei PartProgram

5.1 Upload file

Per poter mandare un partprogram tramite MQTT ad Alleantia è necessario inoltrare un messaggio JSON tipo quello mostrato in esempio:

```
{
   “component”:”DEVICES”,
   “operation”:”UPLOAD”,
   “devId”: [ 63 ],
   “partProgram”: {
   “requestId”: 1,
   “path”: “ROOT/”,
    “filename”: “prova.txt”,
    “content”:[10,-1,0,9]
}


}
```

Il campo requestID è un identificativo della richiesta arbitrario utile per ritrovare la risposta associata nel topic della telemetry, il content è l’array dei byte del file che si vuole mandare
L’IoT SCADA risponderà con il seguente messaggio:

```
{“accepted”:true,”description”:”requestid 1: prova.txt is upload correctly”,”devSn”:”44eb0bdc5-2b5d-3bb8-9d58-d7a9ccc6ef35”,”onTime”:”Jun 21, 2022 11:30:30 AM”,”ontTimeMillisUTC”:1655803830188}
```

5 Scambio dei PartProgram

5.2 Download file

Per poter scaricare un partprogram tramite MQTT da Alleantia è necessario inoltrare un messaggio JSON tipo quello mostrato in esempio:

```
{
   “component”:”DEVICES”,
   “operation”:”DOWNLOAD”,
   “devId”: [ 63 ],
   “partProgram”: {
   “requestId”: 10,
   “path”: “ROOT/prova.txt”
}


}
```

L’IoT SCADA risponderà con il seguente messaggio:

```
{“devices”:[{“devId”:194,”filename”:”prova.txt”,”content”:[10,-1,0,9]}
```

5.3 ListFile

Per poter avere la lista dei partprogram tramite MQTT da Alleantia è necessario inoltrare un messaggio JSON tipo quello mostrato in esempio:

```
{
   “component”:”DEVICES”,
   “operation”:”LISTFILES”,
   “devId”: [ 194 ],
   “partProgram”: {
   “requestId”: 1,
   “path”: “ROOT/”
}


}
```

L'IoT SCADA risonderà con il seguente messaggio:

{"devices":[{"devId":194,"isFolder":true,"path":"ROOT/
sub/","length":0},{"devId":194,"isFolder":false,"path":"ROOT/prova.txt","length":4}],"de
vSn":"44eb0bdc5-2b5d-3bb8-9d58-d7a9ccc6ef35","onTime":"Jun 20, 2022 3:22:43
PM","ontTimeMillisUTC":1655731363436}

5.4 Delete

Per poter cancellare un partprogram tramite MQTT il messaggio JSON tipo è quello mostrato in esempio:

```
{
   "component":"DEVICES",
   "operation":"DELETE",
   "devId": [ 194 ],
   "partProgram": {
   "requestId": 1,
   "path": "ROOT/prova.txt"
}


}
```

L'IoT SCADA risponderà con il seguente messaggio:

{"accepted":true,"description":"requestId 1: ROOT/prova.txt is deleted
correctly","devSn":"44eb0bdc5-2b5d-3bb8-9d58-d7a9ccc6ef35","onTime":"Jun 23, 2022 10:25:00
AM","ontTimeMillisUTC":1655972700623}

5 Scambio dei PartProgram

5.5 Create SubFolder

Per poter creare una nuova folder per i partprogram tramite MQTT il messaggio JSON tipo è mostratoin esempio:

```
{
   “component”:”DEVICES”,
   “operation”:”CREATESUBFOLDER”,
   “devId”: [ 194 ],
   “partProgram”: {
   “requestId”: 1,
   “path”: “ROOT/”,
   “dirname”:”sub”
}


}
```

L’IoT SCADA risponderà con il seguente messaggio:

```
{“accepted”:true,”description”:”requestId 1: the folder ROOT/sub is created correctly”,”devSn”:”44eb0bdc5-2b5d-3bb8-9d58-d7a9ccc6ef35”,”onTime”:”Jun 23, 2022 10:26:10 AM”,”ontTimeMillisUTC”:1655972770346}
```

5 Scambio dei Part Program

5.6 Delete SubFolder

Per poter cancellare una subfolder il messaggio JSON tipo da inoltrare è questo:

```
{
   “component”:”DEVICES”,
   “operation”:”DELETESUBFOLDER”,
   “devId”: [ 194 ],
   “partProgram”: {
   “requestId”: 1,
   “path”: “ROOT/sub”
}


}
```

L’IoT SCADA risponderà con il seguente messaggio:

```
{“accepted”:true,”description”:”requestId 1: the folder ROOT/sub is deleted correctly”,”devSn”:”44eb0bdc5-2b5d-3bb8-9d58-d7a9ccc6ef35”,”onTime”:”Jun 23, 2022 10:29:09 AM”,”ontTimeMillisUTC”:1655972949642}
```

6 Informazioni sugli allarmi

6.1 Configurazioni degli allarmi

Per ottenere tali informazioni il messaggio JSON da inoltrare è:

```
{
   “component”:”ALARMS”,
   “operation”:”CONFIG”,
   “varId”: [ 47, 48]    //OPZIONALE
}
```

Il parametro lista varId è facoltativo e serve per filtrare le informazioni: omettendo tale parametro il dispositivo fornirà le informazioni di tutti gli allarmi. Impostandolo, verranno restituite solamente le informazioni di quelli selezionati.

L’IoT Scada risponderà con una serie di informazioni relative agli eventi:

```
{
  “alarmConfigList”: [
    {
      “id”: 48,
      “description”: “ALLARME CONTROTESTA”,
      “condition”: “$G_63_86 eq true”
    },
    {
      “id”: 43,
      “description”: “ALL.UTENSILI MOTORIZZATI”,
      “condition”: “$G_63_81 eq true”
    },
    {
      “id”: 47,
      “description”: “ALLARME CARICATORE”,
      “condition”: “$G_63_85 eq true”
    }

],
  “devSn”: “ IOTSPIXXXXXXXXX”,
  “onTime”: “Feb 20, 2017 5:25:48 PM”
}
```

6 Informazioni sugli allarmi

6.2 Stato degli allarmi

Per ottenere tali informazioni il messaggio JSON da inoltrare è:

```
{
   “component”:”ALARMS”,
   “operation”:”DATA”,
   “varId”: [ 47, 48]    //OPZIONALE
}
```

Il parametro lista varId è facoltativo e serve per filtrare le informazioni: omettendo tale parametro il dispositivo fornirà le informazioni di tutti gli allarmi. Impostandolo, verranno restituite solamente le informazioni di quelli selezionati.

L’IoT Scada risponderà con una serie di informazioni relative agli allarmi:

```
{
  “alarmDataList”: [
    {
      “id”: 48,
      “quality”: true,
      “alarmed”: false
    },
    {
      “id”: 43,
      “quality”: true,
      “alarmed”: true
    },
    {
      “id”: 47,
      “quality”: false,
      “alarmed”: false
    }
],
  “devSn”: “ IOTSPIXXXXXXXXX”,
  “onTime”: “Feb 20, 2017 5:25:48 PM”
}
```

7 Informazioni sugli eventi

7.1 Configurazioni degli eventi

Per ottenere tali informazioni il messaggio JSON da inoltrare è:

```
{
   “component”:”EVENTS”,
   “operation”:”INFO”,
   “varId”: [ 1 ]    //OPZIONALE
}
```

Il parametro lista varId è facoltativo e serve per filtrare le informazioni: omettendo tale parametro il dispositivo fornirà le informazioni di tutti gli eventi. Impostandolo, verranno restituite solamente le informazioni di quelli selezionati.

L’IoT Scada risponderà con una serie di informazioni relative agli eventi:

```
{
  “eventsInfoList”: [
    {
      “eventId”: 1,
      “eventName”: “Nuovo Evento”,
      “type”: “boolean”,
      “condition”: “$G_63_71”,
      “snapshotGlobalIds”: “G_63_21”,
      “comparisonOperator”: “eq”,
      “numericCompareValue”: 1
    }
  ],
  “devSn”: “ IOTSPIXXXXXXXXX”,
  “onTime”: “Feb 20, 2017 5:25:48 PM”
}
```

7 Informazioni sugli eventi

7.2 Dati storici sugli eventi

Per ottenere tali informazioniil messaggio JSON da inoltrare è:

```
{
   “component”:”EVENTS”,
   “operation”:”HISTORY”,
   “varId”: [ 1 ],       //OBBLIGATORIO
   “startTime”:0,     //OPZIONALE
   “endTime”:123456789     //OPZIONALE
}
```

Il parametro lista varId è obbligatorio. Dovrà esserci un solo valore nella lista. I parametri startTime e endTime sono facoltativi. Tali parametri applicano un filtro temporale sui dati da restituire. I valori da inserire sono espressi in millisecondi a partire dal 1/1/1070 0:00 GMT

L’IOT SCADA risponderà con una serie di informazioni relative agli eventi:

```
{
  “eventHistoryList”: [
    {
      “eventId”: 1,
      “eventName”: “Nuovo Evento”,
      “timestamp”: “Sep 11, 2017 11:00:58 AM”,
      “state”: true,
      “variablesSnapshot”: [
        {
          “devId”: 63,
          “varId”: 21,
          “value”: “//CNC_MEM/USER/PATH1/TECNO”,
          “quality”: true
        }
      ]
    },
    {
      “eventId”: 1,
      “eventName”: “Nuovo Evento”,
      “timestamp”: “Sep 11, 2017 11:11:28 AM”,
      “state”: true,
      “variablesSnapshot”: [
        {
          “devId”: 63,
          “varId”: 21,
         “value”: “//CNC_MEM/USER/PATH1/O9110”,
          “quality”: true
        }
      ]
    },
  ],
```

7 Altri messaggi automatici


```
  “devSn”: “ IOTSPIXXXXXXXXX”,
  “onTime”: “Feb 20, 2017 5:25:48 PM”
}
```

8 Altri messaggi automatici

Una volta attivato il servizio MQTT sull'Iot Scada, questo potrà inoltrare automaticamente verso il broker, oltre ovviamente ai messaggi di telemetria, messaggi relativi a nuovi allarmi e/o eventi, nel momento in cui questi si verificano.
Ovviamente come per le variabili, per far si che il servizio li possa notificare, anche eventuali allarmi personalizzati e/o eventi dovranno prima essere abilitati all'inoltro nell'interfaccia dell'IoT Scada.
Le sintassi dei messaggi inoltrati automaticamente sono le seguenti.

8.1 Nuovo allarme sollevato

In caso di un nuovo allarme, verrà inoltrato verso il topic configurato per la ricezione di allarmi il seguente messaggio JSON:

```
{
  "activeAlarmsList": [
    {
      "id": 11,
      "eventId": 16153,
      "deviceName": "Fanuc 545",
      "measure": "RIPARO APERTO",
      "description": "RIPARO APERTO",
      "onDate": "Sep 22, 2017 10:08:09 AM",
      "offDate": "Sep 22, 2017 10:10:25 AM"
    }
  ],
  "devSn": " IOTSPIXXXXXXXXX",
  "onTime": "Feb 20, 2017 5:25:48 PM"
}
```

Il messaggio verrà inoltrato due volte: nel momento dell'attivazione dell'allarme, e sarà quindi ovviamente assente il parametro offDate, e nel momento del suo rientro, e sarà compilato come esattamente come sopra.

Il messaggio conterrà inoltre il campo **deviceSection** se presente.

8 Altri messaggi automatici

8.2 Nuovo evento sollevato

In caso di un uovo evento, verrà inoltrato verso il topic configurato per la ricezione di eventi il seguente messaggio JSON:


```
{
  “newEventsList”: [
    {
      “eventId”: 1,
      “eventName”: “Nuovo Evento”,
      “type”: “boolean”,
      “condition”: “$G_63_71”,
      “snapshotGlobalIds”: “G_63_21,G_63_820”,
      “comparisonOperator”: “eq”,
      “numericCompareValue”: 1,
      “timestamp”: “Sep 21, 2017 2:58:49 PM”,
      “snapshotVarsDatas”: [
        {
          “globalId”: “G_63_21”,
          “snapshotValue”: “//CNC_MEM/USER/PATH1/TECNO”
        },
        {
          “globalId”: “G_63_820”,
          “snapshotValue”: false
        }
      ],
      “eventValue”: “true”
    }
  ],
  “devSn”: “ IOTSPIXXXXXXXXX”,
  “onTime”: “Feb 20, 2017 5:25:48 PM”
}
```


Esistono due tipologie di eventi: boolean, che sono scatenati al verificarsi di una condizione, e onChange, scatenati al cambiamento di valore di una variabile.
Per la tipologia di evento boolean verranno inoltrati due messaggi verso il broker: uno nel momento in cui la condizione si verifica, un altro nel momento in cui la condizione risulta non più vera.
Gli eventi di tipo onChange, per loro natura, inoltreranno un messaggio verso il broker ogni qualvolta la variabile che monitorano cambia di valore.

Note


Alleantia
ALL <THINGS> JOINED

Alleantia s.r.l.
www.alleantia.com

Sede legale: Via Tosco Romagnola, 136 - 56025 Pontedera (PI)
Sede operativa: Via G.Malasoma 26, 56121 Pisa
Partita IVA/Cod. fiscale: IT 02011550502

Tel: (+39) 050 9911933

In ragione dell'evoluzione delle Norme, dei materiali e delle tecnologie, le caratteristiche riportate nei testi e nelle illustrazioni del presente documento si potranno ritenere impegnative solo dopo conferma da parte di Alleantia s.r.l.

M-MQTT-0622-IT