

Case Study

Dokumentacja

Wykonane przez:

Kacper Grodzicki, Piotr Dębicki

Spis treści

Połączenie z urządzeniem	2
Sposób uruchomienia aplikacji	2
Sposób połączenia z serwerem	2
Sposób i częstotliwość odczytu i zapisu danych oraz wywoływania węzłów-metod ...	2
Konfiguracja agenta	3
D2C messages	3
Device Twin	5
Direct methods.....	6
Kalkulacje i logika biznesowa.....	9
Kalkulacje	9
Logika biznesowa	10

Połączenie z urządzeniem

Sposób uruchomienia aplikacji

Agent to aplikacja konsolowa, którą uruchamia się bezpośrednio z poziomu terminala lub wiersza poleceń, jak każdą inną aplikację konsolową.

Sposób połączenia z serwerem

Agent pobiera z pliku konfiguracyjnego dane OpcServer i używając zawartego w nim adresu, łączy się z serwerem OPC UA.

Sposób i częstotliwość odczytu i zapisu danych oraz wywoływania węzłów-metod

Dane są odczytywane i zapisywane co 5 sekund. W momencie odczytu Agent tworzy tablicę z wszystkimi danymi, które pobiera z węzłów i przeprowadza na nich operacje.

Wywoływane są również metody EmergencyStop i ResetErrorStatus zdefiniowane na serwerze, obsługiwane jako Direct Methods.

Konfiguracja agenta

Agent komunikuje się z serwerem OPC UA za pomocą klienta dostarczonego w bibliotece Opc.UaFx oraz z IoT Hub poprzez bibliotekę Microsoft.Azure.Devices.

Dane niezbędne do połączenia się z serwerem oraz IoT Hub zostały zapisane w pliku konfiguracyjnym **application-properties.json**, zawierający:

- **DeviceConnectionString** – zawiera PrimaryConnectionString, umożliwiający połączenie klienta z urządzeniem w IoT Hub.
- **OpcServer** – zawiera adres serwera OPC UA maszyny.

```
{  
  "DeviceConnectionString": "CONNECTION_STRING",  
  "OpcServer": "OPCSERVERCONNECTION"  
}
```

D2C messages

Agent wysyła 2 rodzaje wiadomości do IoT Hub. Są to dane telemetryczne oraz błędy. Wiadomości telemetryczne są przesyłane co 5 sekund w formacie JSON oraz zawierają następujące dane:

- DeviceId – zawiera nazwę danego urządzenia,
- ProductionStatus – zawiera informacje czy dane urządzenie pracuje,
- WorkerId – zawiera identyfikator aktualnie pracującej maszyny,
- GoodCount – zawiera liczbę wyprodukowanych elementów dobrej jakości,
- BadCount - zawiera liczbę wyprodukowanych elementów złej jakości,
- Temperature – zawiera temperaturę maszyny.

Wed May 21 2025 22:12:50 GMT+0200 (czas środkowoeuropejski letni):

```
{
  "body": {
    "DeviceId": "Device 1",
    "ProductionStatus": 1,
    "WorkerId": "0f54c96f-df79-4eb3-be24-dd1f73e66920",
    "GoodCount": 116,
    "BadCount": 9,
    "Temperature": 65.20461911235685
  },
  "enqueuedTime": "Wed May 21 2025 22:12:50 GMT+0200 (czas środkowoeuropejski letni)"
}
```

Wed May 21 2025 22:12:45 GMT+0200 (czas środkowoeuropejski letni):

```
{
  "body": {
    "DeviceId": "Device 3",
    "ProductionStatus": 1,
    "WorkerId": "3f7db8b1-4ff0-460c-9dad-2ee2e3913554",
    "GoodCount": 100,
    "BadCount": 13,
    "Temperature": 81.89968522740341
  },
  "enqueuedTime": "Wed May 21 2025 22:12:45 GMT+0200 (czas środkowoeuropejski letni)"
}
```

Wed May 21 2025 22:12:45 GMT+0200 (czas środkowoeuropejski letni):

```
{
  "body": {
    "DeviceId": "Device 2",
    "ProductionStatus": 1,
    "WorkerId": "cf056d8a-a3c3-4851-874b-2319713982df",
    "GoodCount": 24,
    "BadCount": 11,
    "Temperature": 945
  },
  "enqueuedTime": "Wed May 21 2025 22:12:45 GMT+0200 (czas środkowoeuropejski letni)"
}
```

Wiadomości z błędami są przysyłane również w formacie JSON gdy nastąpi zmiana stanu błędów maszyny. Zawiera następujące dane:

- DeviceId – zawiera nazwę danego urządzenia,
- Errors – zawiera listę błędów maszyny,
- Value – zawiera liczbową wartość błędów w zapisie dziesiętnym,
- Timestamp – zawiera czas wystąpienia wiadomości.

```
{
  "body": {
    "DeviceId": "Device 3",
    "Errors": [
      "Sensor Failure",
      "Unknown"
    ],
    "Value": 12,
    "Timestamp": "2025-05-21T22:37:16.9252131+02:00"
  },
  "enqueuedTime": "Wed May 21 2025 22:37:17 GMT+0200 (czas środkowoeuropejski letni)"
}
```

Device Twin

Device Twin w Agencie zawiera dane na temat urządzenia w dwóch sekcjach:

1. Desired

Sekcja **Desired** przechowuje ustawienia, które są oczekiwane od urządzenia.

Agent odczytuje parametr **ProductionRate**, który określa oczekiwaną szybkość produkcji dla danej maszyny.

2. Reported

Sekcja **Reported** zawiera dane zgłaszane przez urządzenie w czasie rzeczywistym.

Agent przesyła do IoT Hub m.in. następujące informacje:

- **ProductionRate** – rzeczywista prędkość produkcji raportowana przez urządzenie,
- **DeviceErrors** – liczba błędów wykrytych przez urządzenie.

Device twin You can add tags and desired properties to your device twin here. To remove a tag or desired property, set the value of the item to be removed to 'null'.

```
17  "properties": {
18    "desired": {
19      "Device/1": {
20        "ProductionRate": 0
21      },
22      "Device/2": {
23        "ProductionRate": 0
24      },
25      "$metadata": {
26        "$lastUpdated": "2025-05-19T10:34:59.1238553Z",
27        "$lastUpdatedVersion": 20,
28        "TestProp1": {
29          "$lastUpdated": "2025-05-19T10:34:59.1238553Z",
30          "$lastUpdatedVersion": 20
31        },
32        "testProp6": {
33          "$lastUpdated": "2025-05-19T10:34:59.1238553Z",
34          "$lastUpdatedVersion": 20
35        },
36        "Device/1": {
37          "$lastUpdated": "2025-05-19T10:34:59.1238553Z",
38          "$lastUpdatedVersion": 20,
39          "ProductionRate": {
40            "$lastUpdated": "2025-05-19T10:34:59.1238553Z",
41            "$lastUpdatedVersion": 20
42          }
43        },
44        "Device/2": {
45          "$lastUpdated": "2025-05-19T10:34:59.1238553Z",
46          "$lastUpdatedVersion": 20,
47          "ProductionRate": {
48            "$lastUpdated": "2025-05-19T10:34:59.1238553Z",
49            "$lastUpdatedVersion": 20
50          }
51        }
52      }
53    }
54  }
```

Device twin You can add tags and desired properties to your device twin here. To remove a tag or desired property, set the value o

```
55  "reported": {
56    "DateTimeLastAppLaunch": "2025-05-18T23:35:55.4738862+02:00",
57    "DateTimeLastDesiredPropertyChangeReceived": "2025-05-19T00:07:26.9298386+02:00",
58    "DeviceErrors": 1,
59    "ProductionRate": 0,
60    "Device/1": {
61      "DateTimeLastAppLaunch": "2025-05-21T19:54:43.580126+02:00",
62      "DeviceErrors": 4,
63      "ProductionRate": 0,
64      "DateTimeLastDesiredPropertyChangeReceived": "2025-05-19T12:34:59.2996653+02:00"
65    },
66    "Device/2": {
67      "DateTimeLastAppLaunch": "2025-05-21T19:54:43.9007789+02:00",
68      "DeviceErrors": 12,
69      "ProductionRate": 10,
70      "DateTimeLastDesiredPropertyChangeReceived": "2025-05-19T12:34:59.4685722+02:00"
71    },
72  }
```

Direct methods

Agent obsługuje dwie metody sterujące pracą urządzeń:

1. EmergencyStop

Metoda ta ustawia wartość węzła EmergencyStop na true, tym samym zatrzymując natychmiastowo pracę urządzenia w sytuacjach awaryjnych.

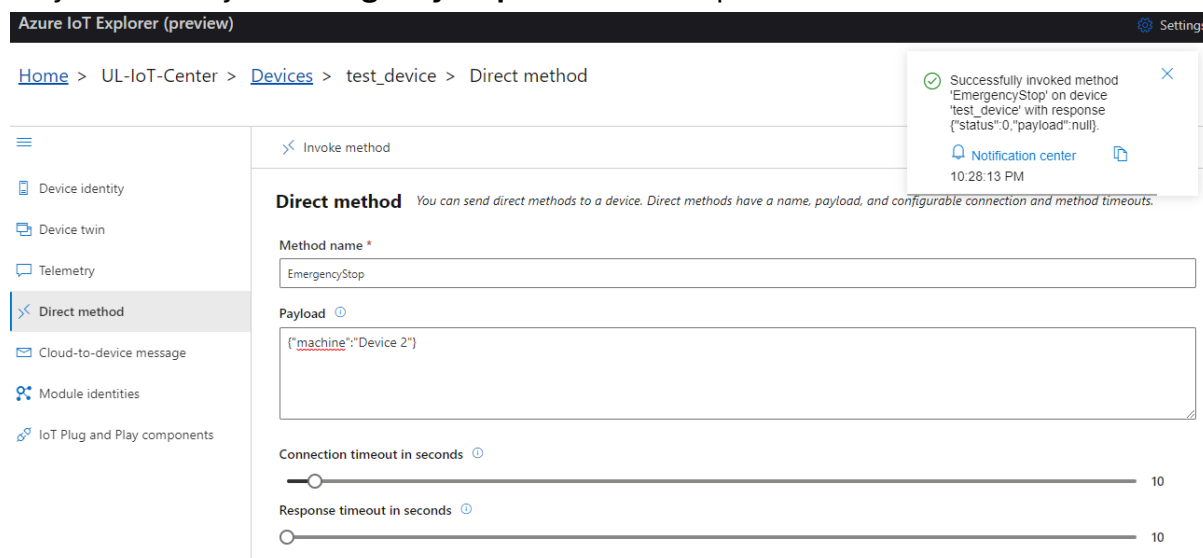
2. ResetErrorStatus

Metoda ta ustawia wartość węzła DeviceErrors na 0, co skutkuje zresetowaniem wszystkich błędów zgłoszonych przez urządzenie.

Aby wywołać jedną z powyższych metod, musimy przekazać nazwę docelowego urządzenia w formacie JSON jako parametr. Przykładowy parametr (payload) dla urządzenia o nazwie **Device 1** wygląda następująco: **{ "machine": "Device 1" }**

Po otrzymaniu żądania wywołania jednej z metod, Agent odczytuje wartość pola "machine" z przesłanego JSON-a i na tej podstawie lokalizuje odpowiedni węzeł w strukturze serwera OPC UA. Następnie wywołuje odpowiednią metodę z użyciem funkcji **CallMethod**, dostępnej w bibliotece Opc.UaFx.

Przykładowe użycie **EmergencyStop** w Azure IoT Explorer.



The screenshot shows the Azure IoT Explorer (preview) interface. The breadcrumb navigation is: Home > UL-IoT-Center > Devices > test_device > Direct method. The left sidebar contains a menu with options: Device identity, Device twin, Telemetry, Direct method (selected), Cloud-to-device message, Module identities, and IoT Plug and Play components. The main area is titled 'Invoke method' and 'Direct method'. It includes a description: 'You can send direct methods to a device. Direct methods have a name, payload, and configurable connection and method timeouts.' The 'Method name' field is set to 'EmergencyStop'. The 'Payload' field contains the JSON string: '{"machine":"Device 2"}'. Below the payload field are two sliders for 'Connection timeout in seconds' and 'Response timeout in seconds', both set to 10. A notification bubble in the top right corner states: 'Successfully invoked method 'EmergencyStop' on device 'test_device' with response {"status":0,"payload":null}. Notification center 10:28:13 PM'.

Industrial Device Simulator

New Device

Remove Selected

Device 1

Device 2

Device 3

Device 4

☐ Production Status

Start

Stop

Production Rate:

20

+

-

Workorder ID:

6e4c35c7-fe3a-40f0-b326-25ab61400

Temperature:

25

Good Count:

617

Bad Count:

322

☒ Emergency Stop

☐ Power Failure
☐ Sensor Failure
☐ Unknown

Przykładowe użycie **ResetErrorStatus** w Azure IoT Explorer

[Home](#) >
[UL-IoT-Center](#) >
[Devices](#) >
[test_device](#) >
Direct method

Device identity

Device twin

Telemetry

Direct method

Cloud-to-device message

Module identities

IoT Plug and Play components

Invoke method

Direct method

You can send direct methods to a device. Direct methods have a name, payload, and configurable connection and method timeouts.

Method name *

ResetErrorStatus

Payload

["machine":"Device 2"]

Connection timeout in seconds

10

Response timeout in seconds

10

Successfully invoked method 'ResetErrorStatus' on device 'test_device' with response {"status":0,"payload":null}.

Notification center

10:29:07 PM

8

Kalkulacje i logika biznesowa

Kalkulacje

W usłudze Azure Stream Analytics zostały zaimplementowane trzy zapytania, które przetwarzają dane przychodzące z urządzeń IoT.

1. Obliczanie wskaźnika jakości produkcji (Production KPIs)

To zapytanie oblicza procentową jakość produkcji dla każdego urządzenia, analizując liczbę dobrych i złych produktów w pięciominutowym oknie czasowym.

2. Monitorowanie temperatury urządzeń

To zapytanie agreguje dane temperatury dla każdego urządzenia w przesuwającym się oknie czasowym. Wyliczane są wartości maksymalne, minimalne oraz średnie.

3. Wykrywanie nadmiernych błędów urządzeń

To zapytanie analizuje dane z błędami przesyłane przez osobny route i wykrywa sytuacje, w których liczba błędów przekracza 3 w czasie jednej minuty.

Logika biznesowa

W IoT Hub zostały utworzone 3 funkcję (Azure Function):

1. Automatyczne wyzwolenie Emergency Stop (EmergencyStopFunction)

Funkcja odpowiada za natychmiastowe zatrzymanie urządzenia, gdy wykrytych zostanie więcej niż 3 błędy w ciągu jednej minuty.

Funkcja jest wyzwalana metodą HTTP POST. Oczekuje na dane zawierające identyfikator urządzenia z IoT Hub (DeviceId) oraz nazwę maszyny w świecie rzeczywistym (TargetMachine).

Po otrzymaniu żądania wysyła polecenie bezpośrednie (Direct Method) o nazwie EmergencyStop do TargetMachine.

2. Aktualizacja tempa produkcji (UpdateProductionRateFunction)

Funkcja odpowiada za zmniejszenie oczekiwanego tempa produkcji urządzenia w momencie gdy jego procentowa jakość produkcji wynosi poniżej 90%.

Funkcja jest wyzwalana metodą HTTP POST. Oczekuje na dane zawierające identyfikator urządzenia z IoT Hub (DeviceId), nazwę maszyny w świecie rzeczywistym (TargetMachine) oraz procentowa jakość produkcji (productionKpis).

Po otrzymaniu danych, funkcja sprawdza czy $productionKpis < 90$ i w takim przypadku zmniejsza desired ProductionRate urządzenia DeviceId o 10. W przeciwnym przypadku funkcja zakończy się, nie wykonując żadnych zmian.

3. Wysłanie powiadomień e-mail (SendEmailProxy)

Funkcja odpowiada za powiadamianie operatora e-mailowo, gdy tylko wystąpi **jakikolwiek błąd urządzenia**.

Funkcja jest wyzwalana metodą HTTP POST. Przyjmuje ona dane (DeviceId, czas wystąpienia błędu oraz lista błędów), a następnie przekazuje je do **Azure Logic App**, gdzie realizowane jest wysłanie e-maila na wcześniej zdefiniowany adres.