Centralny System Analityczny

**Instrukcja Bramy IoT**

Wersja 0.2 (dokument w trakcie opracowywania)

Luty, 2017

Centralny System Analityczny

***Brama IoT***

HISTORIA REWIZJI DOKUMENTU

| Wersja | Data | Podsumowanie zmian | Autorzy |
| --- | --- | --- | --- |
| 0.1 | 29-11-2016 |  | Łukasz Swolkień |
| 0.2 | 27-02-2017 |  | Łukasz Swolkień |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Spis Treści

[HISTORIA REWIZJI DOKUMENTU 2](#_Toc468452114)

[WSTĘP 4](#_Toc468452115)

[1.1 Diagram poglądowy 4](#_Toc468452116)

[1.2 Architektura Bramy 5](#_Toc468452117)

[1.3 Główne funkcjonalności 6](#_Toc468452118)

[APLIKACJA 7](#_Toc468452119)

[2.1 Wstęp 7](#_Toc468452120)

[2.2 Logowanie 7](#_Toc468452121)

[2.3 Menu 8](#_Toc468452122)

[2.4 Informacje ogólne 9](#_Toc468452123)

[2.5 Powiadomienia 10](#_Toc468452124)

[2.6 Manager użytkowników 10](#_Toc468452125)

[2.7 Interfejsy 11](#_Toc468452126)

[2.8 Narzędzia 12](#_Toc468452127)

[REST API 14](#_Toc468452128)

[3.1 Uwierzytelnianie 14](#_Toc468452129)

[3.2 Dostęp do zasobów sieci sensorycznej (IoT) 16](#_Toc468452130)

[3.2.1 /v1/sn 16](#_Toc468452131)

[3.2.2 /v1/node-types 16](#_Toc468452132)

[3.2.3 /v1/nodes 17](#_Toc468452133)

[ZASTOSOWANIA 20](#_Toc468452134)

[4.1 Możliwe obszary zastosowań 20](#_Toc468452135)

[ZAŁĄCZNIKI 25](#_Toc468452136)

[5.1 Specyfikacja techniczna urządzeń pomiarowych 25](#_Toc468452137)

[5.1.1 Czujniki wielkości fizycznych 25](#_Toc468452138)

[5.1.2 MCP01 czujniki stanu zaworów kanalizacji podciśnieniowej 25](#_Toc468452139)

[5.1.3 MZA –xxx monitor pracy zaworu kanalizacji podciśnieniowej. 26](#_Toc468452140)

# WSTĘP

1

## Diagram poglądowy

## 1.2 Architektura Bramy

Główną barierą do realizacji zadań sieci czujników jest to, że ponad 85% istniejących urządzeń nie było zaprojektowanych w celu udostępniania danych za pośrednictwem Internetu. Dodatkowo urządzenia nisko-mocowe, ze względu na ograniczone możliwości energetyczne i obliczeniowe, nie mogą bezpośrednio udostępniać danych publicznie. W tym celu znajduje zastosowanie CSA Brama, która pełni funkcję medium łączącego sieć czujników z Internetem przy jednoczesnym zapewnieniu bezpieczeństwa i zdalnego zarządzania zasobami.

**Schemat poglądowy Bramy IoT**



## 1.3 Główne funkcjonalności

1. Umożliwienie modyfikowania ustawień interfejsów LAN oraz WLAN poprzez aplikację sieciową uruchomioną w przeglądarce Chrome lub Firefox
2. Umożliwienie modyfikowania ustawień modułu GSM poprzez aplikację sieciową uruchomioną w przeglądarce Chrome lub Firefox
3. Umożliwienie zdefiniowania parametrów pracy węzła(ów) poprzez aplikację sieciową uruchomioną w przeglądarce Chrome lub Firefox
4. Umożliwienie walidacji połączenia pomiędzy Bramą i urządzeniami pomiarowo-kontrolnymi (węzłami) poprzez aplikację sieciową uruchomioną w przeglądarce Chrome lub Firefox
5. Wizualizacja bieżących i historcznych danych pomiaowych poprzez aplikację sieciową uruchomioną w przeglądarce Chrome lub Firefox
6. Uwierzytelnienie dostępu do odczytu zasobów chronionych lub/i modyfikacji ustawień poprzez aplikację sieciową uruchomioną w przeglądarce Chrome lub Firefox
7. Identyfikacja i rejestrowanie nowego węzła poprzez aplikację sieciową uruchomioną w przeglądarce Chrome lub Firefox
8. Strumieniowanie danych (MQTT, Modbus TCP)
9. REST API

# APLIKACJA

2

## Wstęp

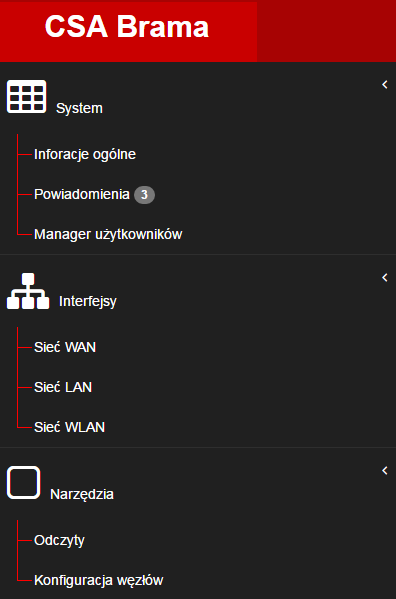
Brama udostępnia aplikacje webową na porcie 8080. Aplikacja może być uruchomiana w przeglądarce Chrome lub Firefox na komputerze, tablecie lub telefonie.

## Logowanie

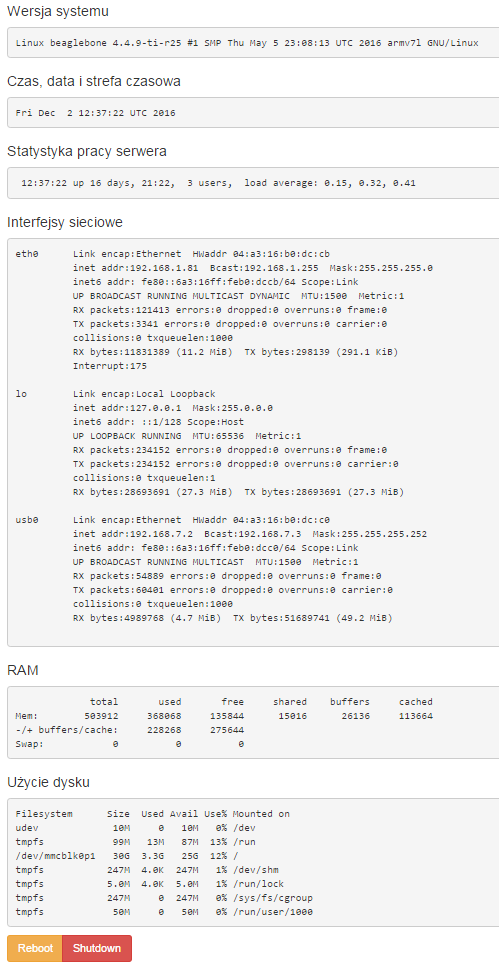


## Menu

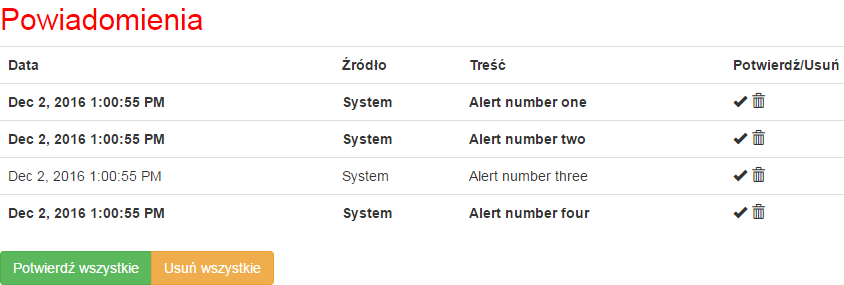
Menu applikacji jest kontekstowe i udostępnia funkcjonalności do których uprawnienia ma zalogowany użytkownik.



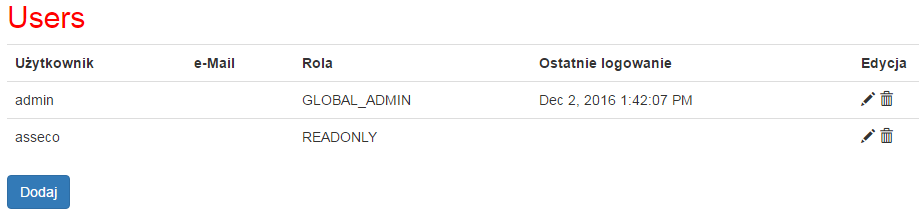
## Informacje ogólne



## Powiadomienia

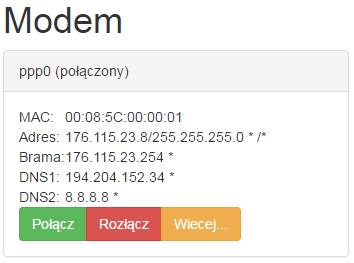


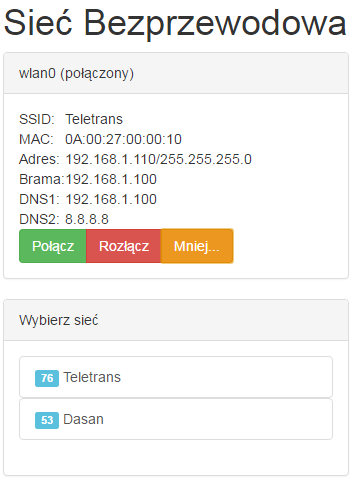
## Manager użytkowników



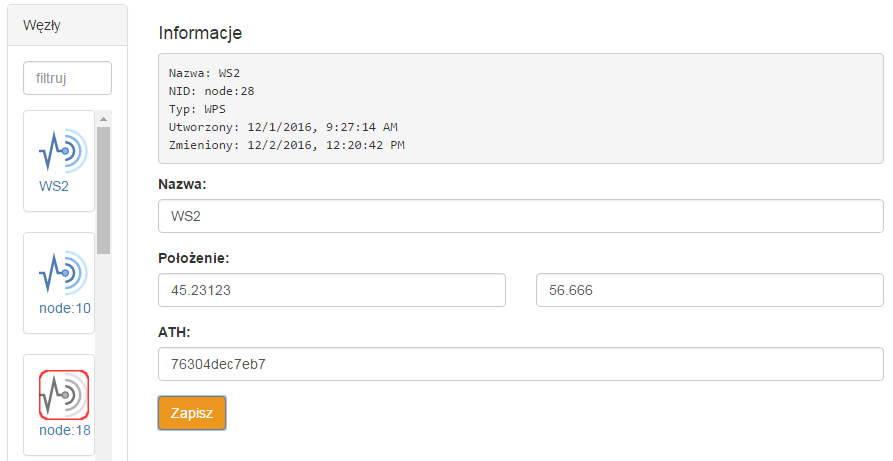
## Interfejsy

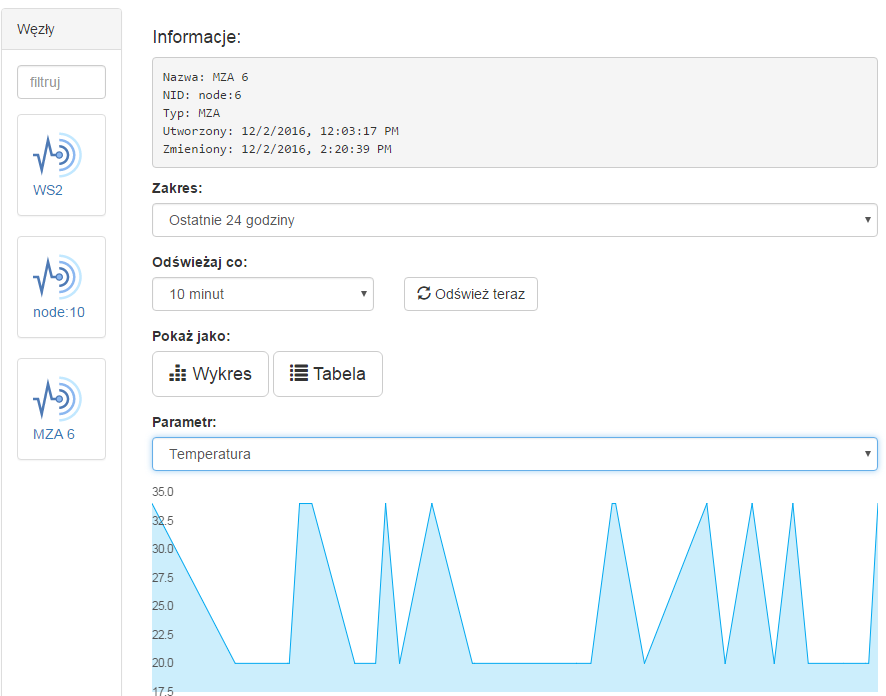






## Narzędzia





# REST API

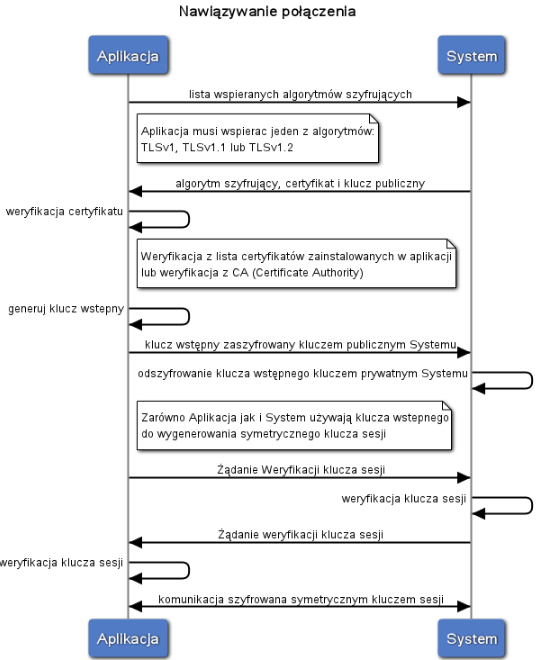
3

## Dostęp do zasobów chronionych

Należy wziąć pod uwagę możliwe cyber-ataki, oraz możliwość dostępu do danych lub sterowania przez osoby niepowołane. W tym przypadku stosuje się odpowiednio mechanizmy: autentykacji, autoryzacji i szyfrowania danych. To sprowadza się do wielu czynności takich jak: ustalanie poziomów dostępności do danych i możliwości ich modyfikacji, następnie ustalanie i zmienianie haseł dostępowych oraz zmienianie kluczy oraz algorytmów szyfrujących.

Autoryzacja, autentykacja i przydzielanie przywilejów dostępu powinno być określane centralnie, aby zagwarantować unikalność przywilejów a co się z tym wiąże brak konfliktów pomiędzy przywilejami tudzież niekontrolowanej zmiany przywilejów w zależności od miejsca dostępu.

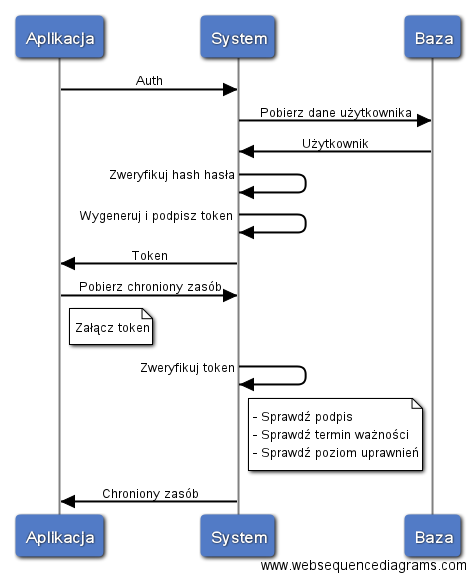
Dostęp do chronionych zasobów systemu obywa się dwuetapowo. Najpierw musi być nawiązane połączenie szyfrowane TLS, a następnie następuje uwierzytelnienie.



W dalszym kroku dostęp do chronionych zasobów systemu oparty będzie o autentykację z wykorzystaniem technologii JSON Web Tokens (JWT) opisanej w standardzie RFC7519 (<https://tools.ietf.org/html/rfc7519>). JWT wykorzystuje tak zwane tokeny, czyli ciągi znaków identyfikujących sesję i potwierdzających prawa posiadacza tokenu (aplikacji klienta) do dostępu do żądanych zasobów. Zaletą technologii JWT są:

* Bezstanowość – informacje o sesji zawarte są w samym tokenie i nie jest wymagane przechowywanie ich po stronie serwera. Eliminuje to problem zarządzania cyklem życia sesji i usuwania wygasłych sesji.
* Bezpieczeństwo – każdy token podpisany jest kluczem kryptograficznym, w tym wypadku bezpiecznym algorytmem HS256.
* Łatwość implementacji – informacje zawarte w tokenie są zwięzłe i oparte o czytelny format JSON.

Typowy przebieg działania aplikacji, która wymaga dostępu do chronionego zasobu systemu, przedstawia poniższy diagram:



Symulacja scenariusza z użyciem komendy curl:

1. Aplikacja wywołuje końcówkę /auth metodą POST, przekazując nazwę użytkownika i hasło w formacie JSON zawartym w treści żądania:

curl -X POST -H "Content-Type: application/json" -d '{"username": "admin", "password" : "abc123"}' <http://192.168.56.2:8080/auth>

1. System, po pozytywnej weryfikacji nazwy użytkownika i hasła zwraca wygenerowany token:

{

"access\_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFkbWluIiwicm9sZSI6IkdMT0JBTF9BRE1JTiIsImV4cCI6MTQ3ODI4NjAyNCwiaWF0IjoxNDc4MTk5NjI0LCJuYmYiOjE0NzgxOTk2MjQsImlkZW50aXR5IjoiYWRtaW4ifQ.IVOIGDt4o9ksMKguZMzLVeRhr5Y\_aYxuPMPHtb2HbY8"

}

Token ten zawiera następujące informacje zakodowane w formacie BASE64:

Nagłówek  
{  
 "alg": "HS256",

"typ": "JWT"

}

Dane

{

"username": "admin",

"role": "GLOBAL\_ADMIN",

"exp": 1478286024,

"iat": 1478199624,

"nbf": 1478199624,

"identity": "admin"

}

Ostatnią część stanowi podpis kryptograficzny.

1. Aplikacja wysyła żądanie do docelowego zasobu, dodając nagłówek *Authorization* zawierający słowo „JWT” i otrzymany token:

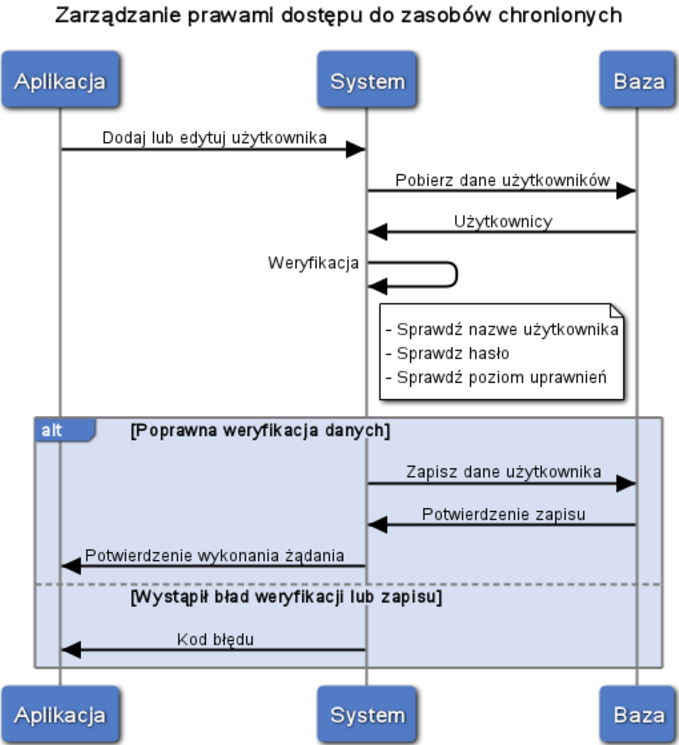
curl -X GET -H 'Authorization: JWT eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFkbWluIiwicm9sZSI6IkdMT0JBTF9BRE1JTiIsImV4cCI6MTQ3ODI4NjAyNCwiaWF0IjoxNDc4MTk5NjI0LCJuYmYiOjE0NzgxOTk2MjQsImlkZW50aXR5IjoiYWRtaW4ifQ.IVOIGDt4o9ksMKguZMzLVeRhr5Y\_aYxuPMPHtb2HbY8' <http://192.168.56.2:8080/api/v1/users/>

## Zarządzanie użytkownikami systemu

W celu zarządzania dostępem do chronionych zasobów należy stworzyć w systemie użytkowników i nadać im odpowiednie role uprawniające do wykonywania odpowiadających im operacji. Wstępnie identyfikujemy podstawowe role potrzebne w systemie wymienione poniżej, niemniej jednak system musi być otwarty na możliwość dodania kolejnych ról bez potrzeby większych zmian w systemie.

GLOBAL\_ADMIN – pełne uprawnienia do dodawania, edycji i usuwania zasobów systemu,  
READONLY – dostęp tylko do odczytu zasobów chronionych

Typowy przebieg działania systemu, która zarządza pobieranie, tworzeniem i edycja uprawnień dla użytkowników, przedstawia poniższy diagram.



Przykładowa odpowiedź wygenerowana żądanie pobrania użytkowników powinna wyglądać następująco:

{

"size": 2,

"users": [

{

"username": "jkowal",

"email": "jkowal@sample.com",

"last\_login": "1477846892688",

"role": "READONLY"

},

{

"username": "admin",

"email": null,

"last\_login": "1487497000106",

"role": "GLOBAL\_ADMIN"

}

]

}

Przykładowe żądanie modyfikacji lub dodania użytkownika powinna wyglądać następująco:

{

"username": "jkowal",

"password": "abc123",

"email": "jkowal@sample.com",

"role": "GLOBAL\_ADMIN"

}

Usuwanie użytkownika z systemu:

DELETE /v1/users/{username}

## Dostęp do zasobów sieci sensorycznej (IoT)

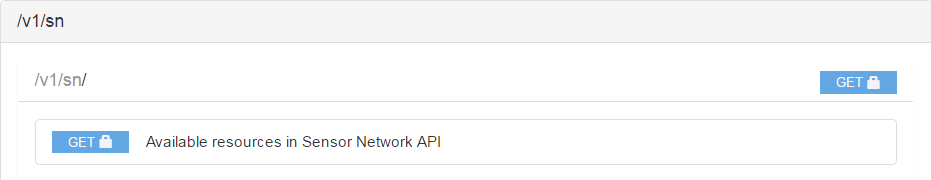
Opis API wraz z przykładami dostępny jest w <http://192.168.7.2:8080/doc> lub/i pliku *api.html*

Poniżej znadują się informacje uzupełniające.

Dostęp do zasobów udostępnionych poprzez infrastrukturę IoT jest chroniony i wymaga uwierzytelnienia oraz autoryzacji. Uwierzytelnianie i autoryzacja dokonywane jest na podstawie tokenu załączonego w nagłówku żądania. Zawiera on również informacje potrzebną do potwierdzenia poziomu uprawnień potrzebnych do uzyskania dostępu do żądanego zasobu.

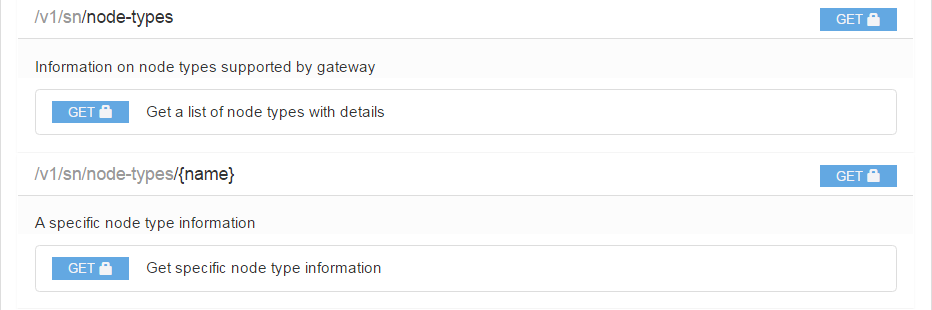
### /v1/sn

Każdy zasób dostarcza metadane (Hypermedia – HATEOAS), które zawierają informację o tym jak można dostać się do kolejnych zasobów. Umożliwia to łatwą nawigację pomiedzy zasobami i usuwa potrzebę ingerencji w istniejący kod aplikacji klienckiej przy jakichkolwiek zmianach w API. Punktem wejscia jest zasób /sn (sn – sensor network)

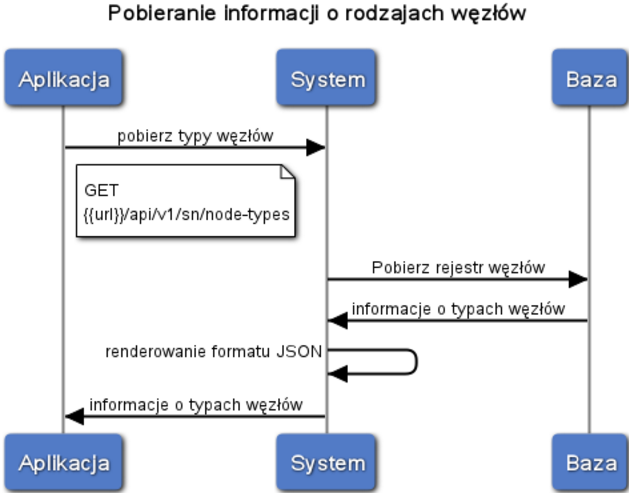


### /v1/node-types

Informacja o wspieranych typach węzłów zawiera szczegółowe informacje o dostępnych atrybutach każdego rodzaju urządzenia IoT i mierzonych przez niego wartości.



Typowy przebieg działania aplikacji, która pobiera dane o rodzajach węzłów, przedstawia poniższy diagram.



Typy węzłów mogą być wykorzystane do automatycznego mapowania danych pomiarowych z strukturami danych agregowanych w systemach nadrzędnych lub przy wiązaniu danych z kontrolkami graficznie przedstawiającymi dane na panelu wizualizacyjnym.

Przykładowa odpowiedź wygenerowana na żądanie /api/v1/sn/node-types wygląda następująco:

{

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/node-types"

},

"nodeTypes": [

{

"fields": [

"nid",

"status",

"battery"

],

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/node-types/BCP"

},

"name": "BCP"

},

{

"fields": [

"nid",

"status",

"cycle",

"opening",

"pressure",

"temperature",

"battery"

],

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/node-types/MZA"

},

"name": "MZA"

}

],

"size": 2

}

### 3.2.3 /v1/nodes

Informacje ogólne węzłów dotyczące:

- ilości węzłów,

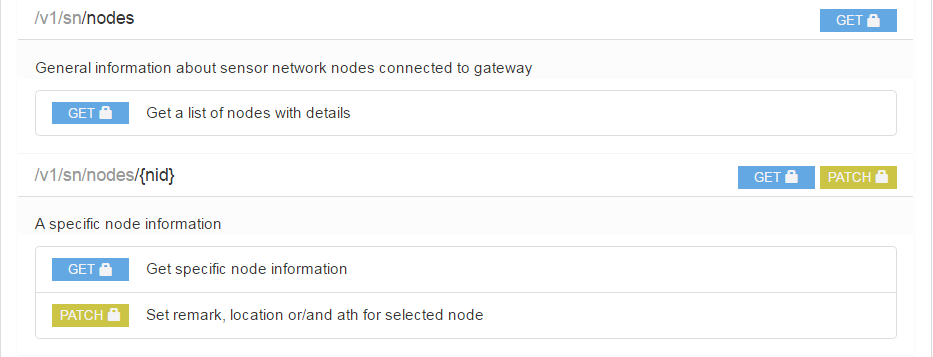
- czasu pierwszej i ostatniej aktualizacji danych dla każdego z węzłów,

- lokalizacja węzła (GPS),

- stanie uwierzytelnienia węzła

- parametry zapytania umożliwiają uzupełnienie odpowiedzi o dodatkowe informacje na temat typu węzła lub/i o dane z ostatniego odczytu (rekordu)

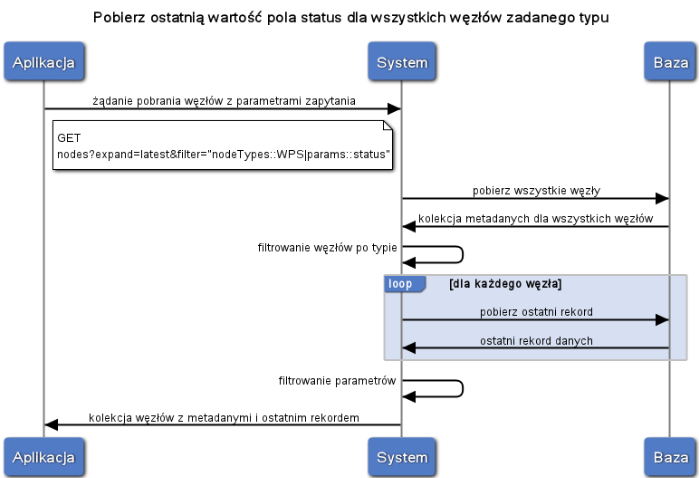
?expand=latest,nodeTypes



Możliwe jest też filtrowanie parametrów danego rodzaju urządzenia:

?filter=”nodeTypes::MZA|parameters::battery,status,cycle”

Typowy przebieg działania aplikacji, która pobiera dane z węzłów danego typu wraz z ostatnimi zapisanymi danymi, przedstawia poniższy diagram.



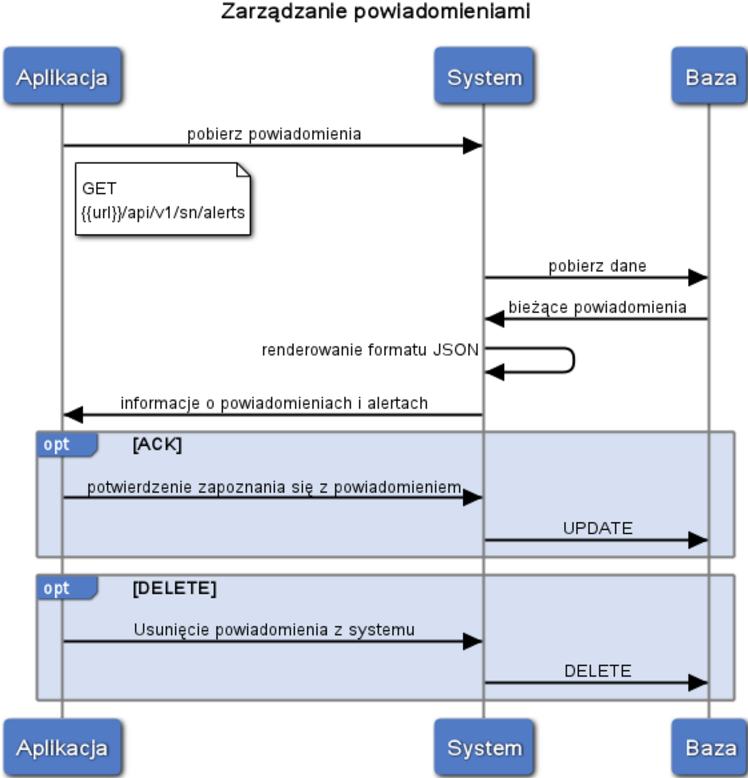
### 3.2.4 /v1/sn/records

{{bridge\_url}}/api/v1/sn/records?filter="interval::2016-12-02T09:00:00,2016-12-02T12:00:00|nodeTypes:MZA,WSP"

Cdn....

Model procesu zarządzania powiadomieniami i alertami

Typowy przebieg działania aplikacji, która zarządza powiadomieniami i alertami z systemu, przedstawia poniższy diagram.



Przykładowa odpowiedź wygenerowana na zadane pobrania powiadomień powinna wyglądać następująco:

{

"meta": { "href": "http://192.168.7.2/api/v1/sn/alerts" },

"alerts": [

{

"ack": false,

"message": "Out of order",

"meta":{ "href": "http://192.168.7.2/api/v1/sn/alerts/dogPzIz8" },

"name": "alert",

"nid": "node:11",

"remark": null,

"severity": 1,

"type": "Defect"

},

{

"ack": false,

"message": "low battery level: 12%",

"meta":{ "href": "http://192.168.7.2/api/v1/sn/alerts/nYrnfYEv" },

"name": "alert",

"nid": "node:10",

"remark": null,

"severity": 2,

"type": "Maintenance"

},

{

"ack": true,

"message": "over 10 days inactive",

"meta":{ "href": "http://192.168.7.2/api/v1/sn/alerts/hwX6aOr7" },

"name": "alert",

"nid": "node:123",

"remark": null,

"severity": 3,

"type": "Maintenance"

}

]

}

Przykładowe żądanie potwierdzenia powiadomieniu lub alertu:

{

"ack": true,

"remark": "some remark",

"severity": 1

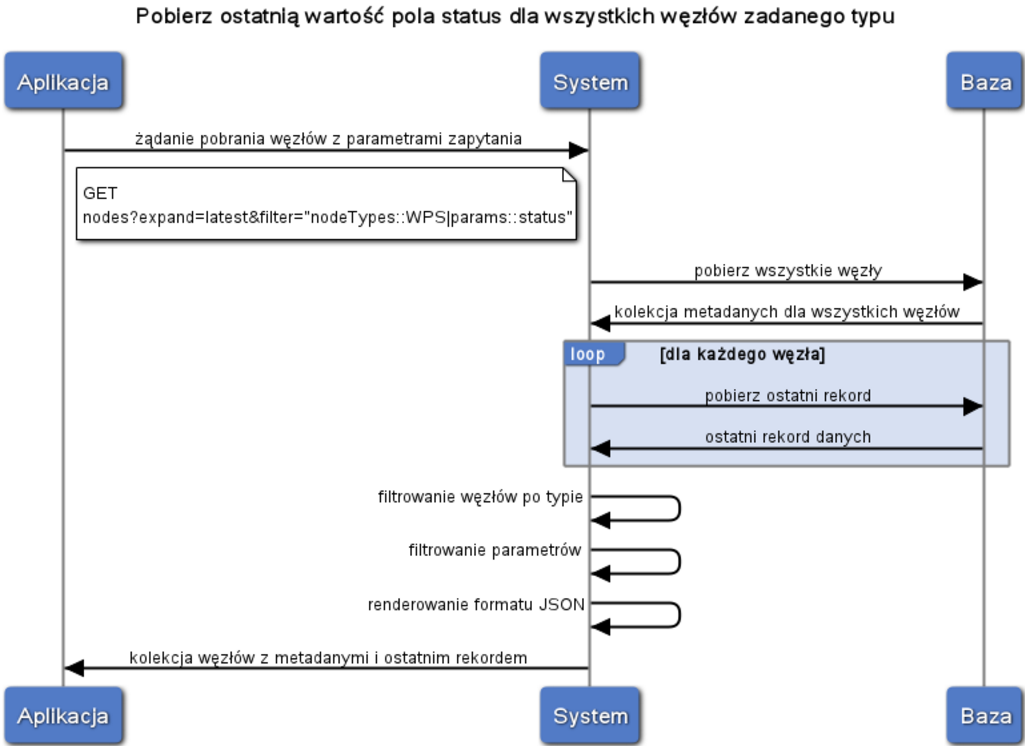
}

Usuwanie powiadomienia lub alertu:

DELETE /v1/sn/alerts/{alert\_id}

Model procesu odczytu aktualnych danych pomiarowych

Typowy przebieg działania aplikacji, która pobiera dane z węzłów danego typu wraz z ostatnimi zapisanymi danymi, przedstawia poniższy diagram.



Przykładowa odpowiedź wygenerowana na żądanie powinna wyglądać następująco:

{

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/nodes"

},

"nodes": [

{

"latest": {

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/records/latest/WPS:9"

},

"record": {

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/records/1487506804"

},

"size": 1,

"timestamp": 1487506804

},

"status": "open"

},

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/nodes/WPS:9"

}

},

{

"latest": {

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/records/latest/WPS:8"

},

"record": {

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/records/1487506017"

},

"size": 1,

"timestamp": 1487506017

},

"status": "closed"

},

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/nodes/WPS:8"

}

},

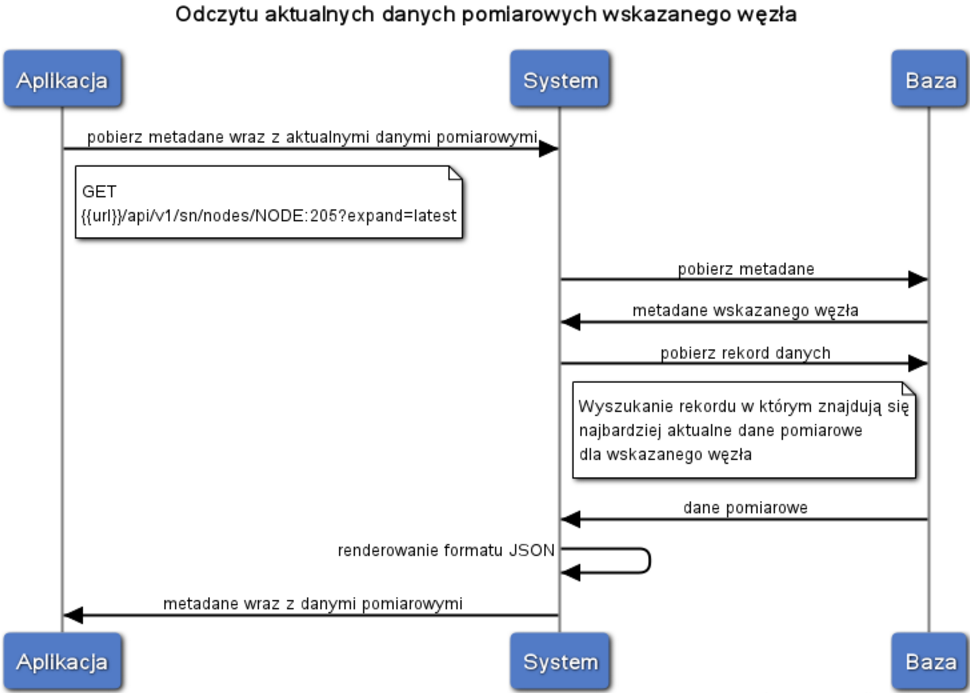
...

],

"size": 9

}

Typowy przebieg działania aplikacji, która pobiera najbardziej aktualne dane pomiarowe dla wskazanego węzła:



Przykładowa odpowiedź wygenerowana na żądanie powinna wyglądać następująco:

"ath": "54c0b946-f69d",

"createdAt": 1485486453,

"latest": {

"battery": 21,

"cycle": 36,

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/records/latest/MZA:205"

},

"nid": "MZA:205",

"opening": 8378,

"pressure": 35,

"record": {

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/records/1487504523"

},

"size": 1,

"timestamp": 1487504523

},

"status": "open",

"temperature": 16

},

"location": null,

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/nodes/MZA:205"

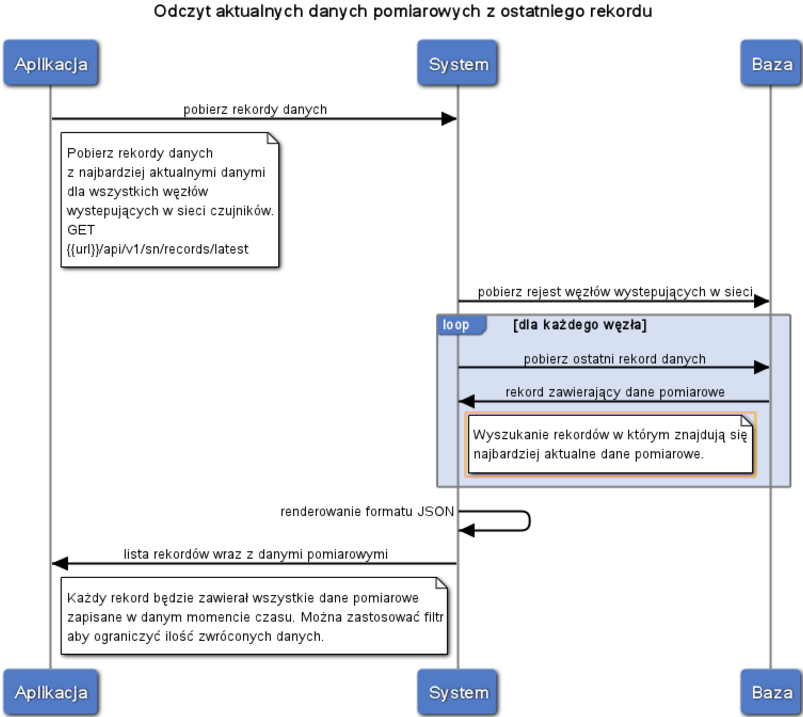
},

"nid": "MZA:205",

"remark": null,

"updatedAt": 1487504523

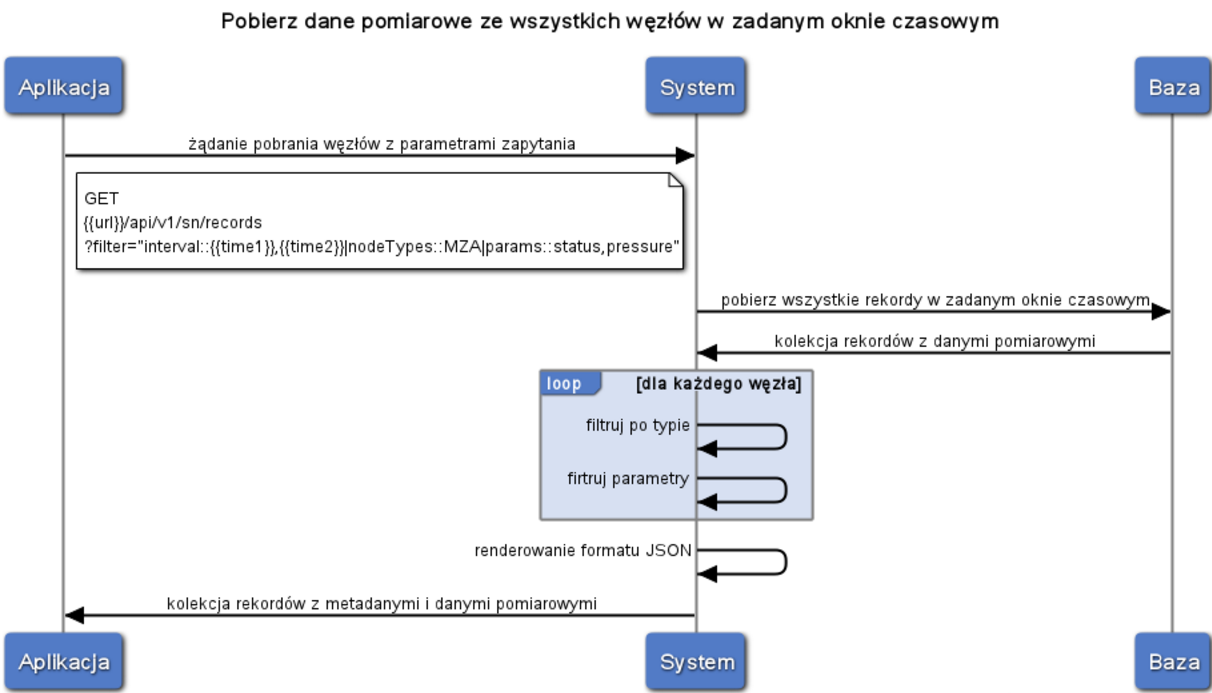
Typowy przebieg działania aplikacji, która pobiera rekordy w których zapisano najbardziej aktualne dane pomiarowe dla wszystkich węzłów:



Warto odnotować, że powyższa metoda znajdzie zastosowanie przede wszystkim w momencie ponownej próby przesłania danych z Bramy do serwera klasy ‘enterprise’. W celu pobrania aktualnych danych dla węzła lub grupy węzłów powinna być wykorzystana jedna z metod opisanych powyżej.

Model odczytu historycznych danych pomiarowych

Typowy przebieg działania aplikacji, która pobiera dane historyczne z zadanego okresu czasowego z wskazaniem węzłów danego typu oraz podzbioru oczekiwanych parametrów, przedstawia poniższy diagram:



Żądanie pobrania danych pomiarowych węzłów wraz z przykładowymi parametrami zapytania powinno wyglądać następująco:

[GET]{{base\_url}}/api/v1/sn/records?filter="interval::{{time1}},{{time2}}|nodeTypes::MZA|params::status,pressure"

gdzie:

{{base\_url}} – adres system udostepniającego zasoby w standardzie REST API

oraz parametry zapytania ?filter=”...” zaznaczone powyżej kolorem żółtym oznaczją odpowiednio:

interval::{{time1}},{{time2}} – zakres zadanego okresu czasowego

noteTypes::MZA,BCP,... – lista typów węzłów

params::status,pressure,... – lista żądanych danych pomiarowych.

Opcje w parametrze ?filter oddzielone sa znakiem: | (pipe)

Parametry zapytania (filter=”...”) są opcjonalne i jeśli nie zostaną podane system zwróci rekordy danych zawierające wszystkie typy węzłów i wszystkie parametry otrzymane na przestrzeni ostatniej godziny.

Przy projektowaniu struktury danych należy przyjąć najmniejszą „rozdzielczość” z jaką agregowane będą dane otrzymane z sieci czujników. Dla większości zastosowań powinna wystarczyć rozdzielczość jedno sekundowa. W przypadku większej ilości pomiarów przesłanych do bramy w ciągu jednej sekundy wszystkie dane zostaną zapisane w jednym rekordzie. Nie jest to duże ograniczenie gdyż dla różnych czujników wszystkie dane zostaną prawidłowo zapisane i jedynie dane przesłane przez ten sam czujnik w ciągu jednej sekundy zastaną nadpisane. W sieciach czujników nisko-mocowych przypadek tak częstej transmisji z tego samego czujnika jest wykluczony z założenia.

Przykładowa odpowiedź wygenerowana na żądanie powinna wyglądać następująco:

{

"meta": { "href": "http://192.168.7.2/api/v1/sn/records" },

"size": 2,

"records": [

{

"meta": { "href": "http://192.168.7.2/api/v1/sn/records/1478330706" },

"timestamp": 1478330706,

"size": 2,

"nodes": [

{

"meta": { "href": "http://192.168.7.2/api/v1/sn/nodes/MZA:1" },

"nid": "MZA:1",

"battery": 20,

"cycle": 100,

"opening": 45,

"pressure": 90,

"status": "open",

"temperature": 25,

"nodeType": {

"meta": { "href": "http://192.168.7.2/api/v1/sn/node-types/MZA" },

"name": "MZA",

"fields": ["nid", "status", "cycle", "opening", "pressure", "temperature", "battery"]

}

},

{

"battery": 93,

"meta": { "href": "http://192.168.7.2:8080/api/v1/sn/nodes/WPS:6" },

"nid": "WPS:6",

"status": "closed",

"nodeType": {

"meta": { "href": "http://192.168.7.2/api/v1/sn/node-types/WPS" },

"name": "WPS",

"fields": ["nid", "status", "battery]

}

}

]

},

{

"meta": { "href": "http://192.168.7.2/api/v1/sn/records/1478330736" },

"timestamp": 1478330736,

"size": 1,

"nodes": [

{

"meta": { "href": "http://192.168.7.2/api/v1/sn/nodes/MZA:1" },

"nid": "MZA:1",

"battery": 24,

"cycle": 12323,

"opening": 1332,

"pressure": 2834,

"status": "closed",

"temperature": 50,

"nodeType": {

"meta": { "href": "http://192.168.7.2/api/v1/sn/node-types/MZA" },

"name": "MZA",

"fields": ["nid", "status", "cycle", "opening", "pressure", "temperature", "battery"]

}

}

]

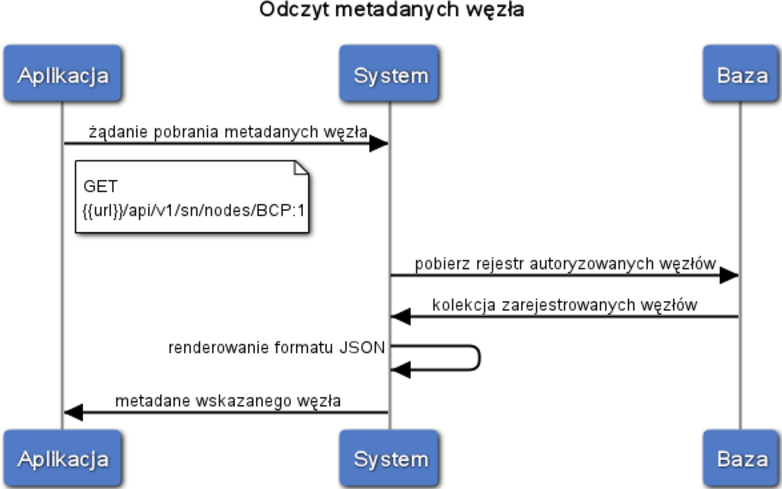
}

]

}

Model odczytu metadanych węzła

Typowy przebieg działania aplikacji, która pobiera metadane wskazanego węzła, przedstawia poniższy diagram:



{

"meta": {

"href": "http://192.168.7.2:8080/api/v1/sn/nodes/BCP:1"

},

"ath": "92361f002671",

"createdAt": 1485484114,

"updatedAt": 1487430127,

"nid": "BCP:1",

"remark": null,

"location": {

"lat": 50.032597,

"lng": 19.938678

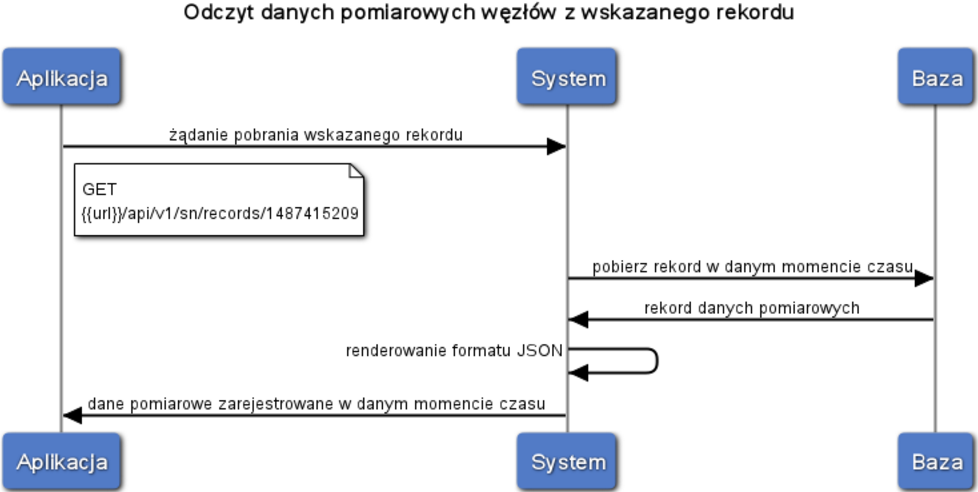
}

}

Przykładowa odpowiedź w formacie JSON powinna wyglądać następująco:

Model odczytu danych pomiarowych węzłów z wskazanego rekordu

Typowy przebieg działania aplikacji, która pobiera dane pomiarowe z wskazanego rekordu z zadanego momentu czasu, przedstawia poniższy diagram:



Przykładowa odpowiedź w formacie JSON powinna wyglądać następująco:

{

"meta": { "href": "http://192.168.7.2/api/v1/sn/records/1487415209" },

"timestamp": 1487415209,

"size": 2,

"nodes": [

{

"meta": { "href": "http://192.168.7.2/api/v1/sn/nodes/MZA:1" },

"nid": "MZA:1",

"battery": 20,

"cycle": 100,

"opening": 45,

"pressure": 90,

"status": "open",

"temperature": 25,

"nodeType": {

"meta": { "href": "http://192.168.7.2/api/v1/sn/node-types/MZA" },

"name": "MZA",

"fields": ["nid", "status", "cycle", "opening", "pressure", "temperature", "battery"]

}

},

{

"battery": 93,

"meta": { "href": "http://192.168.7.2:8080/api/v1/sn/nodes/BCP:6" },

"nid": "BCP:6",

"status": "closed",

"nodeType": {

"meta": { "href": "http://192.168.7.2/api/v1/sn/node-types/BCP" },

"name": "BCP",

"fields": ["nid", "status", "battery"]

}

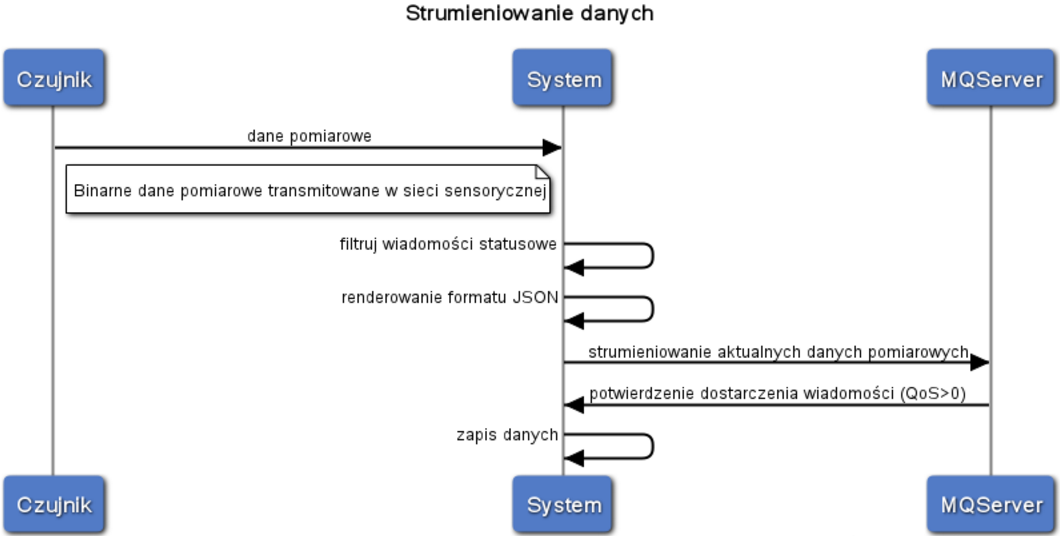
}

]

}

Model zdarzeniowy udostępniania danych

Typowy przebieg działania aplikacji, która przesyła wiadomość w formacie JSON w reakcji na na zdarzenie otrzymania aktualnych danych pomiarowych, przedstawia poniższy diagram:



Przykładowa odpowiedź w formacie JSON generowana przez System na zdarzenie otrzymania aktualnych danych pomiarowych:

{

"remark": null,

"data\_type": "MZA",

"address": 0,

"location": null,

"ath": null,

"nid": "MZA:202",

"updatedAt": 1487432276,

"createdAt": 1485486209,

"latest": {

"status": "open",

"temperature": 41,

"opening": 34,

"battery": 100,

"nid": "MZA:202",

"record": {

"timestamp": 1487432276

},

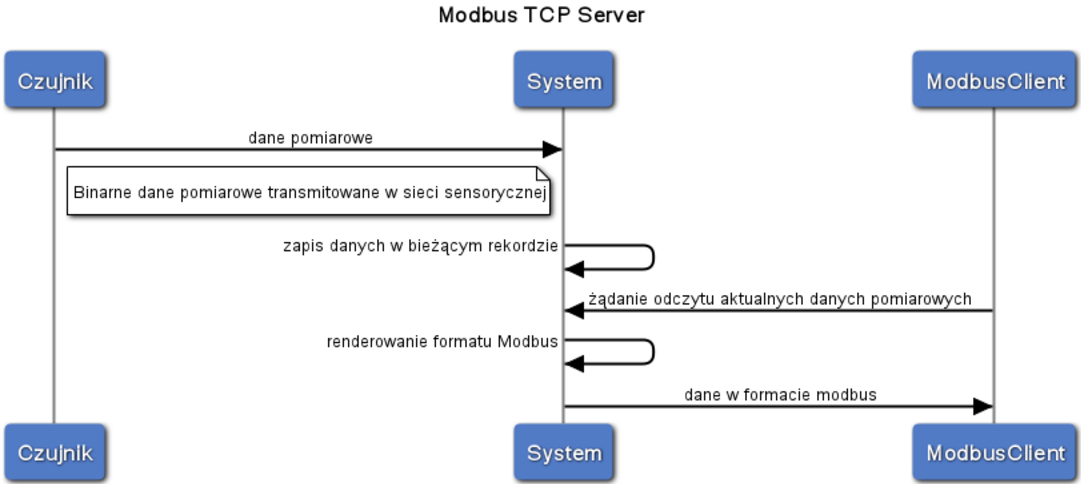
"pressure": 245,

"cycle": 36

}

}

Typowy przebieg działania aplikacji, która udostępnia dane w formacie MODBUS w reakcji na zdarzenie otrzymania aktualnych danych pomiarowych, przedstawia poniższy diagram:



Modbus TCP Server zapisuje dane pomiarowe w kontekście i udostępnia je dla urządzenia odbiorczego (slave device) w tzw „Read Holding Registers, Function Code=0x03”, czyli w analogowych rejestrach wyjściowych. Rezerwowana jest stała przestrzeń 20 rejestrów na potrzeby reprezentacji danych pomiarowych. Przykładowo:

x00 x03 x50 x05 x01 x01 x64 x2A x4A x15 x34 x00 x00 x00 x00 x00 x00 x00 x00 x00 x00 x00

gdzie:

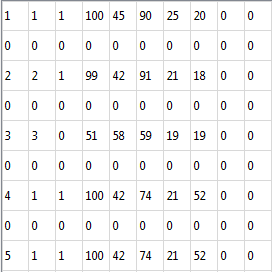
x00 – slave id

x03 – FC=03

x50 – adres, gdzie adres jest liczony następująco: 0x14 \* data[0] - 0x14 if data[0] > 0 else 0x00. Data[0] zawiera informacje o ID węzła.

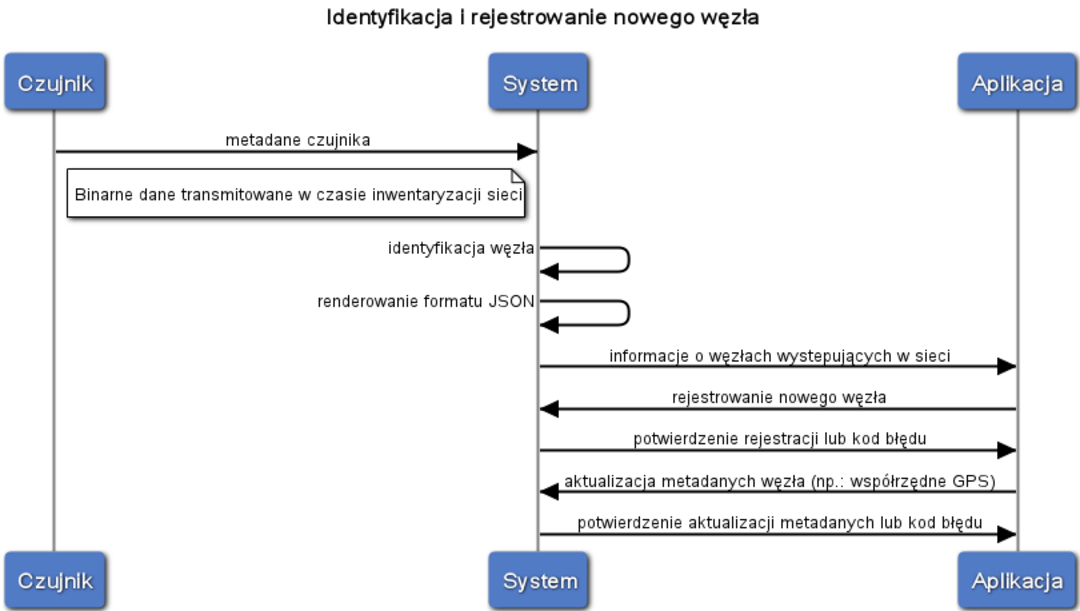
x05 x01 x01 x64 x2A x4A x15 x34 x00 x00 x00 x00 x00 x00 x00 x00 x00 x00 x00 – rejestry zawierające dane pomiarowe.

Przykładowa macierz rejestrów z aktualnymi danymi pomiarowych generowana przez Modbus TCP Server:



Model identyfikacji i rejestrowania nowego węzła

Typowy przebieg działania sieci, systemu oraz aplikacji administracyjnej, przy inwentaryzacji sieci czujników, przedstawia poniższy diagram:



Przykład modyfikacji metadanych czujnika:

{

"remark": "new-remark",

"location": {

"lat": 50.030800,

"lng": 19.937800

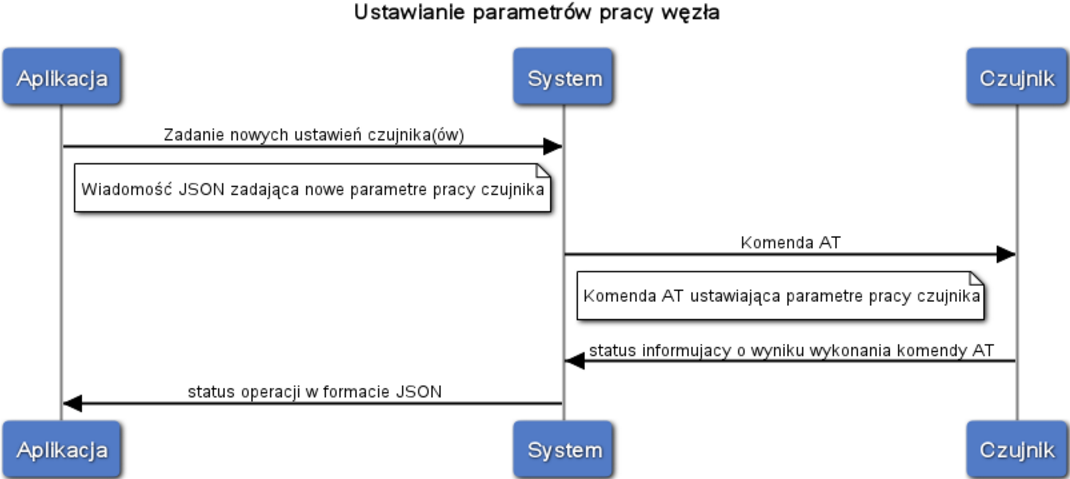
},

"ath": True

}

Model mechanizmu ustawiania parametrów pracy węzła

Typowy przebieg działania aplikacji, przy ustawianiu parametrów pracy węzłów w sieci czujników, przedstawia poniższy diagram:

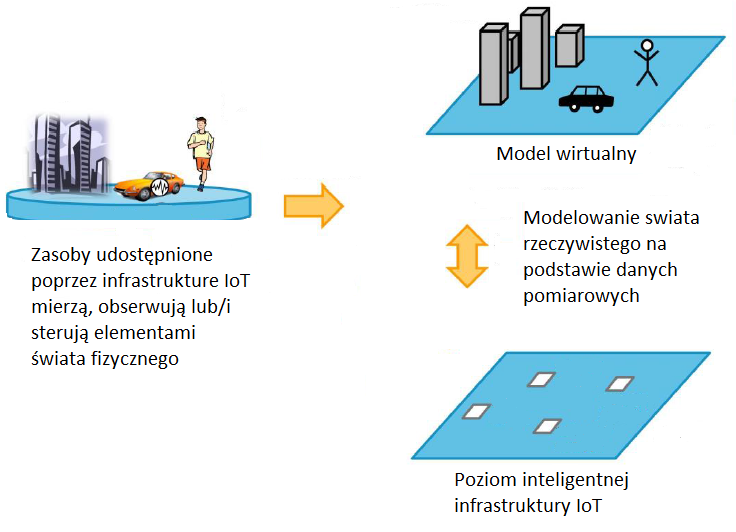


# ZASTOSOWANIA

4

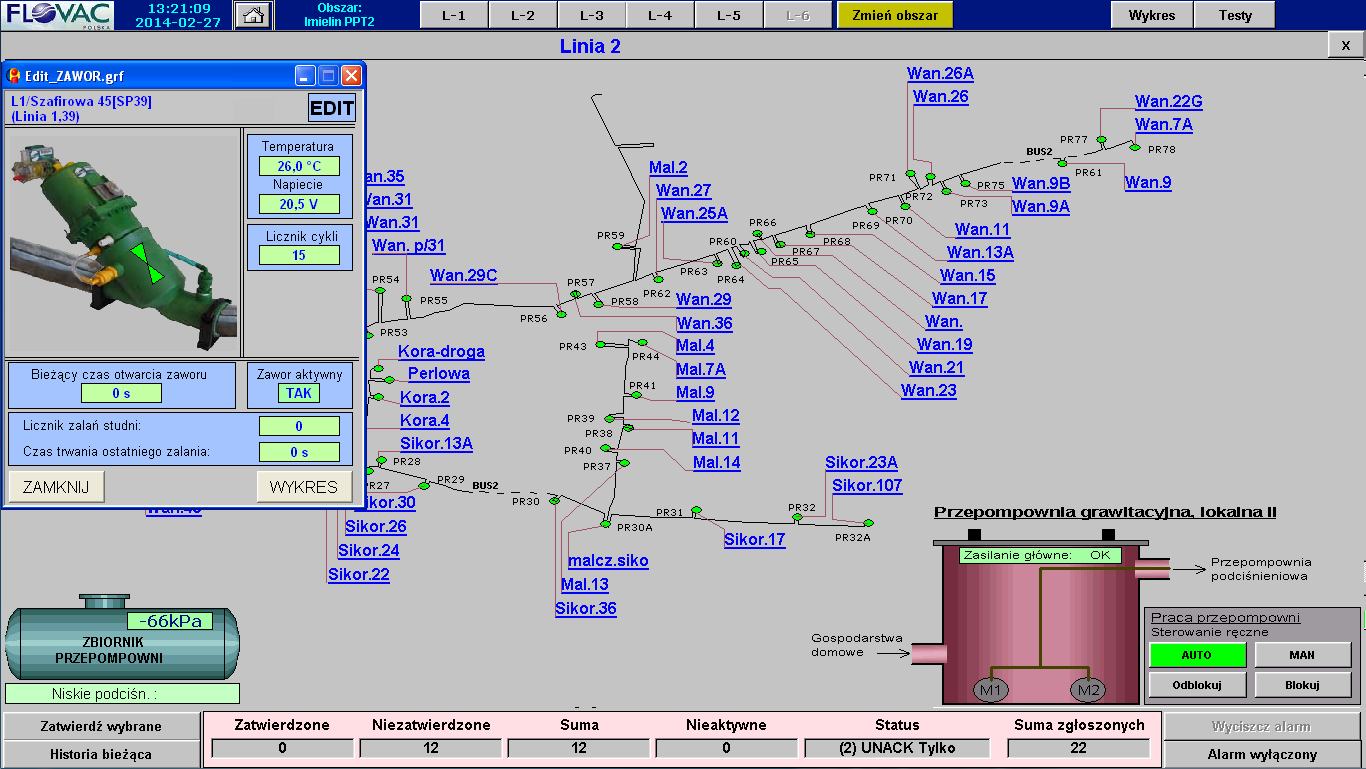
## Możliwe obszary zastosowań

IoT dotyczy każdej dziedziny: przemysł, handel, medycyna, fitness, inteligentne budynki, inteligentne miasta, usługi i inne.



Poniżej przedstawiamy przykładowe obszary w których opracowana przez nas technologia znajduje zastosowanie:

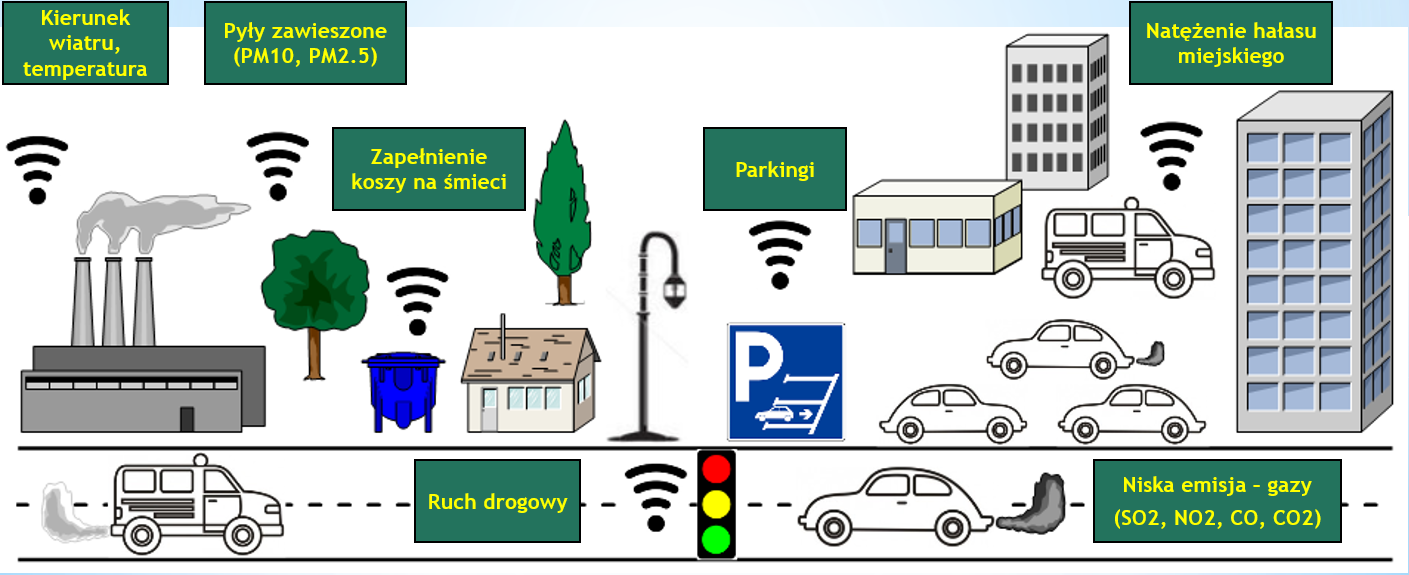
* System bezprzewodowego monitoringu zaworów Flovac - alternatywa dla produkowanego przez naszą firmę systemu monitoringu kanalizacji podciśnieniowej opartego o transmisję kablową,



* Monitoring studzienek wodociągowych i lokalnych wymiennikowni ciepła,
* Bezprzewodowy system monitorowania parametrów sprężonego powietrza dla zakładowych gniazd produkcyjnych w Silgan Whiate Cap S.A.



* Zastosowania w strukturze inteligentnego miasta (smart city):
  + zarządzanie parkingami
  + kontrola pojemników na odpady
  + kontrola parametrów środowiska
    - pomiary niskiej emisji (gazy CO2, CO, SO2, NOx oraz pyły zawieszone PM10, PM2.5) pomiar kierunku wiatru (wpływ na rozkład smogu w mieście)
    - natężenie hałasu miejskiego
    - detekcja gazów w zamkniętych przestrzeniach publicznych



* Instalacje energetyczne z optymalizacją zużycia energii
  + Energia cieplna
  + Energia elektryczna
* Przemysłowe linie technologiczne
* Inteligentne budynki
* Automatyka domowa
* Logistyka
* Transport
* Medycyna
* „Social Machines” Urządzenia współdzielące dane z społecznościami w celu wymiany wiedzy i doświadczeń, np.:
  + Umożliwiające optymalizacje zużycia energii w gospodarstwie domowym
  + Informujące o stanie powietrza w danym regionie
  + Wymieniające dane epidemiologiczne, informujące o intensywności występowania danej choroby zakaźnej na określonym obszarze etc...

# ZAŁĄCZNIKI

5

## Specyfikacja techniczna urządzeń pomiarowych

### Czujniki wielkości fizycznych

Położenie, temperatura, ciśnienie, wilgotność, parametry jakości powietrza, wody itp. Pozwalają na pomiary wartości analogowych w zakresie 0-2,5 V (możliwy pomiar sygnału 4-20mA )

### MCP01 czujniki stanu zaworów kanalizacji podciśnieniowej

MCP01 monitorować ma stan zaworu (otwarty/zamknięty). Będzie przystosowany do pracy w trudnych warunkach występujących w studzienkach kanalizacyjnych. Obudowa MCP-01 wykonana zostanie z materiału ABS i przystosowana do montażu na korpusie zaworu. Konstrukcja musi być odporna na agresywne substancje chemiczne.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Zakładane dane techniczne** | **Typowo** | **Min** | **Max.** | **Jednostka** |
| Napięcie zasilania | 3,6 | 2,5 | 5,5 | [VDC] |
| Pobór prądu: praca | 15 | 8 | 23 | [uA] |
| Wyjście typu otwarty dren |  |  |  |  |
| Rd w obwodzie drenu |  | 10 | 100 | [kom] |
| Wykrywana wielkość szczeliny | ≥5mm\* |  |  |  |
| Temperatura pracy |  | -25 | +50 | [°C] |
| Temperatura przechowywania |  | -30 | +70 | [°C] |
| Wilgotność |  | 0 | 100% |  |

### MZA –xxx monitor pracy zaworu kanalizacji podciśnieniowej.

MZA-xxx służy do monitorowania pracy zaworu instalacji podciśnieniowej sygnalizuje stan zaworu otwarty/zamknięty, czas otwarcia, temperaturę otoczenia, wykonuje statystyki cykli pracy zaworu. Dane przesyła do układu nadrzędnego magistralą dwuprzewodową to znaczy dane i energia zasilająca wykorzystują tą samą parę przewodów.