Spis treści

[Generowanie PPE 2](#_Toc173496065)

[Struktura komendy 2](#_Toc173496066)

[Struktura danych wejściowych 2](#_Toc173496067)

[Plik konfiguracyjny 2](#_Toc173496068)

[Plik z profilem standardowym 4](#_Toc173496069)

[Opis działania programu 5](#_Toc173496070)

[Struktura danych wyjściowych 5](#_Toc173496071)

[Generowanie profili 6](#_Toc173496072)

[Struktura komendy 6](#_Toc173496073)

[Struktura danych wejściowych 6](#_Toc173496074)

[Opis działania programu 6](#_Toc173496075)

[Walidacja 7](#_Toc173496076)

[Struktura komendy 7](#_Toc173496077)

[Struktura danych wejściowych 7](#_Toc173496078)

[Opis działania programu 7](#_Toc173496079)

# Generowanie PPE

Funkcja ta pozwala generować zbiór punktów pomiarowych z odpowiednio do nich przypisanymi profilami standardowymi i produktami energetycznymi na podstawie pliku.

## Struktura komendy

generatorcsire.py generuj-ppe **-i** PLIK\_KONFIG [-s PLIK\_PROFIL] **-o** PLIK\_WYNIKOWY

-i PLIK\_KONFIG Nazwa pliku konfiguracyjnego

-s PLIK\_PROFIL Nazwa pliku z profilem standardowym, domyślnie profil\_standardowy.xlsx

-o PLIK\_WYNIKOWY Nazwa pliku wynikowego

Przykład użycia:

generatorcsire.py generuj-ppe -i Klasy-PP.xlsx -o DanePPE.csv

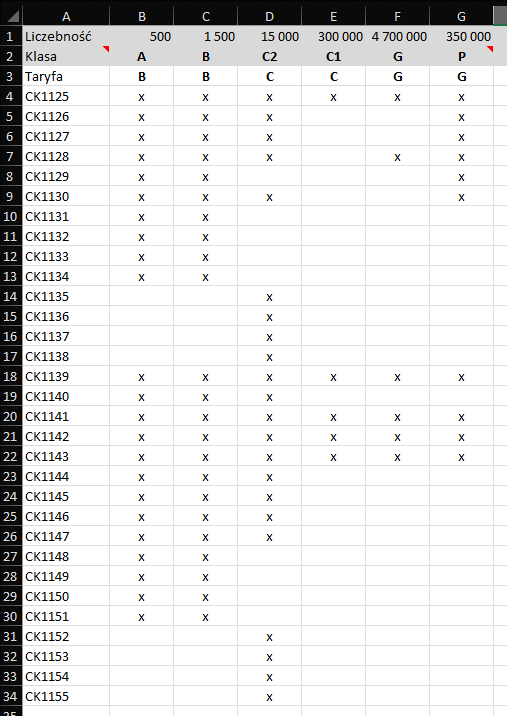
## Struktura danych wejściowych

### Plik konfiguracyjny

Plik konfiguracyjny musi być plikiem w formacie .xlsx, zawierać 7 kolumn i 34 wiersze

Plik ten przedstawia liczebność, grupę taryfy i produkty energetyczne punktów pomiarowych z danej klasy.

Pierwszy wiersz tego pliku zawiera liczebności odpowiednich klas których nazwy zawarte są w wierszu drugim. Trzeci wiersz zawiera informację o tym z jakiej grupy taryfę wybrać dla danej klasy. W kolejnych wierszach wymienione są typy produktów energetycznych. Jeżeli w komórce na przecięciu wiersza z kodem produktu energetycznego i kolumny z nazwą klasy znajduje się krzyżyk, oznacza to że dana klasa ma zawierać dany produkt energetyczny.



Rysunek 1 Wzorcowy plik konfiguracyjny

### Plik z profilem standardowym

Plik z profilem standardowym musi być plikiem w formacie .xlsx, każdy z arkuszy ma zawierać 27 kolumn i 367 wierszy.

Obraz zawierający zrzut ekranu, Równolegle, Prostokąt, kwadrat

Opis wygenerowany automatycznie

Rysunek 2 Wzorcowy plik z profilem standardowym

Każdy z arkuszy pliku odpowiada danej taryfie więc musi zostać nazwany zgodnie z jej nazwą.

W kolumnie pierwszej arkusza zawarta jest nazwa dnia tygodnia dla daty obok w kolumnie drugiej. Kolejne kolumny odpowiadają kolejnym godzinom doby. Dodatkowa kolumna na końcu jest wypełniana dla dni w których nastąpiła zmiana czasu.

## Opis działania programu

Na podstawie pliku konfiguracyjnego generowane jest tyle punktów pomiarowych danej klasy ile wynosi jej liczebność. Każdy punkt ma przypisany swój 18 znakowy kod PPE generowany zgodnie z GS1. Struktura kodu PPE GS1:

5905435S1S2S3S4S5S6S7S8S9S10K,

gdzie:

590 – oznacza prefiks polskiej organizacji GS1

5435 – numer OSD

S1–10 – unikalny identyfikator nadany przez spółkę dla danego PPE

K – cyfra kontrolna

Z użyciem pliku z profilami standardowymi tworzona jest lista taryf możliwych do przypisania dla danej klasy na podstawie informacji o grupie taryfy. Z tej listy taryfa dla poszczególnego punktu wybierana jest losowo. Zapisywana jest informacja o klasie danego punktu oraz tworzona jest lista kodów produktów energetycznych odpowiednich dla tej klasy.

## Struktura danych wyjściowych

Dane wyjściowe dla każdego punktu zawierają kod PPE, taryfę, klasę oraz tablicę produktów energetycznych. Elementy te są oddzielone przecinkami, a dane w tablicy produktów energetycznych oddzielane są dwukropkiem. Dane poszczególnych punktów oddzielone są od siebie znakiem nowej linii.

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Rysunek 3 Przykładowe dane wygenerowane za pomocą programu

# Generowanie profili

Funkcja ta pozwala na podstawie podanej w pliku listy PPE generować dane w formacie .xml według struktury komunikatów 6.1.1.1, 6.1.1.5 i 7.1.1.4

## Struktura komendy

generatorcsire.py generuj-**kodkomunikatu** [-i PLIK\_PPE] [-s PLIK\_PROFIL] **-o** KATALOG\_WYNIKOWY **-d** DOBA [-p PACZKA]

kodkomunikatu Kod komunikatu spośród podanych: {6.1.1.1, 6.1.1.5, 7.1.1.4}

-i PLIK\_PPE Nazwa pliku z ppe, domyślnie ppe.csv

-s PLIK\_PROFIL Nazwa pliku z profilem standardowym, domyślnie profil\_standardowy.xlsx

-o KATALOG\_WYNIKOWY Ścieżka do katalogu wynikowego

-d DOBA Doba w formacie RRRRMMDD

-p PACZKA Wielkość paczki, domyślnie 1000

Przykład użycia:

generatorcsire.py generuj-6.1.1.1 -i DanePPE.csv -o wynik -d 20240801

## Struktura danych wejściowych

Plik z profilem standardowym według tego samego formatu co plik standardowy dla funkcji generuj-ppe. Plik z ppe musi wyglądać jak plik generowany przez funkcję generuj-ppe: Każdy punkt w nowej linii, dane dla punktu oddzielone przecinkami, dane w tablicy produktów energetycznych oddzielone dwukropkiem.

## Opis działania programu

Na podstawie pliku z profilem standardowym, dla każdej taryfy pobierane są informacje o podanej dobie.

Dla każdego produktu energetycznego PPE, na podstawie jego taryfy oraz podstawy dla jego klasy przydzielane jest roczne zużycie wg wzoru:

Losowa liczba w zakresie: (1/5 – 5) \* podstawa

Następnie pomiar obliczany jest na podstawie typu produktu energetycznego. Dla niektórych (straty i napięcia) produktów pomiary są określane jako stała wartość z losowym odchyleniem. Pomiar dodawany jest cztery razy, jako że interwał pomiędzy pomiarami wynosi 15min.

Dane zapisywane są do pliku .xml, po tyle punktów pomiarowych w pliku, ile określono w parametrze paczka. Pliki zapisywane są w podanym w komendzie katalogu wynikowym, który należy uprzednio utworzyć. Nazwa każdego pliku składa się z kodu komunikatu, podanej doby i liczby porządkowej pliku.

# Generowanie wskazań

Funkcja ta pozwala na podstawie podanej w pliku listy PPE generować dane w formacie .xml według struktury komunikatu 6.2.1.1

## Struktura komendy

generatorcsire.py generuj-6.2.1.1 [-i PLIK\_PPE] **-s** PLIK\_KONFIG **-o** KATALOG\_WYNIKOWY [-p PACZKA]

-i PLIK\_PPE Nazwa pliku z ppe, domyślnie ppe.csv

-s PLIK\_PROFIL Nazwa pliku konfiguracyjnego

-o KATALOG\_WYNIKOWY Ścieżka do katalogu wynikowego

-p PACZKA Wielkość paczki, domyślnie 1000

Przykład użycia:

generatorcsire.py generuj-6.2.1.1 -i DanePPE.csv -s Klasy-PP.xlsx -o wynik

## Struktura danych wejściowych

Plik z profilem standardowym według tego samego formatu co plik standardowy dla funkcji generuj-ppe. Plik z ppe musi wyglądać jak plik generowany przez funkcję generuj-ppe: Każdy punkt w nowej linii, dane dla punktu oddzielone przecinkami, dane w tablicy produktów energetycznych oddzielone dwukropkiem.

Plik konfiguracyjny musi być plikiem w formacie .xlsx, zawierać 9 kolumn i 34 wiersze. Dla każdego z produktów energetycznych z 1 kolumny, w 9 kolumnie znajdują się kody typów rejestru pomiarowego licznika możliwych dla danego produktu.

Obraz zawierający tekst, zrzut ekranu, numer, Równolegle

Opis wygenerowany automatycznie

Rysunek 4 Wzorcowy plik konfiguracyjny dla generowania wskazań

## Opis działania programu

Na podstawie pliku konfiguracyjnego tworzona jest lista możliwych typów rejestru dla danego produktu energetycznego. Dla każdego ppe, w zależności od liczby będącej trzecim znakiem kodu jego taryfy (ilość stref), tyle rejestrów licznika zostanie wygenerowanych dla pojedynczego produktu energetycznego tego ppe, oczywiście jeżeli produkt energetyczny ma przypisane jakieś typy rejestrów. Pomiar rejestru licznika jest liczbą losowaną w odpowiednim zakresie. Powód publikacji wybierany jest według prawdopodobieństwa spośród dwóch wartości: CK0130 i CK0131.

# Walidacja

## Struktura komendy

generatorcsire.py waliduj **-i** PLIK [-o PLIK\_WYNIKOWY]

-i PLIK Nazwa pliku lub folderu plików do walidacji

-o PLIK\_WYNIKOWY Nazwa pliku wynikowego

Przykład użycia:

generatorcsire.py waliduj **-i** plik\_1.xml

## Struktura danych wejściowych

Dane wejściowe muszą być plikiem .xml lub folderem zawierającym pliki .xml

## Opis działania programu

Dla każdego pliku w celu zidentyfikowania według którego szablonu XML Schema będzie walidowany, odczytywana jest wartość elementu BusinessProcessMessageType. Szablony odczytywane są z folderu xsdfiles. Jeżeli w komendzie podano nazwę pliku wynikowego, wynik operacji zostanie do niego zapisany.

# Wysyłanie danych do kolejki

## Struktura komendy

generatorcsire.py do-kolejki -i KATALOG -s PLIK\_KONFIG [-n LICZBA\_WATKOW]

-i KATALOG Ścieżka do katalogu z plikami

-s PLIK\_KONFIG Ścieżka do pliku konfiguracyjnego

-n LICZBA\_WATKOW Liczba wątków, domyślnie 4

Przykład użycia:

generatorcsire.py do-kolejki -i wynik -s config.json

## Struktura danych wejściowych

Dane wejściowe muszą być folderem zawierającym pliki .xml. Plik konfiguracyjny musi być w formacie .json i zawierać pola: nazwa użytkownika, hasło, URL brokera, adres kolejki

{  
 "username": "artemis",  
  
 "password": "artemis",  
  
 "brokerurl": "tcp://localhost:61616",  
  
 "queue": "$sys.mqtt.sessions::kolejka"  
}

## Opis działania programu

Każdy plik .xml z katalogu zostaje przesłany do kolejki w oddzielnej wiadomości, do nagłówka wiadomości dodawane jest pole JMSCorrelationID, którego wartością jest tekst z pola MessageId pliku. Informacje o wykonanych operacjach i ewentualnych błędach zapisywane są do pliku log.txt.

# Pobieranie danych z kolejki

## Struktura komendy

generatorcsire.py z-kolejki -s PLIK\_KONFIG -q KOLEJKA -o KATALOG

-s PLIK\_KONFIG Ścieżka do pliku konfiguracyjnego

-q KOLEJKA Adres kolejki

-o KATALOG Ścieżka do katalogu wynikowego

Przykład użycia:

generatorcsire.py z-kolejki -s config.json -q .mqtt.sessions::kolejka -o wynik

## Struktura danych wejściowych

Plik konfiguracyjny musi być w formacie .json i zawierać pola: nazwa użytkownika, hasło, URL brokera.

## Opis działania programu

Program pobiera wszystkie wiadomości z kolejki o podanym adresie, działa aż do manualnego zatrzymania. Wiadomości zapisywane są w oddzielnych plikach .xml których nazwę stanowi MessageId otrzymanej wiadomości.