

## **SPECYFIKACJA TECHNICZNA**

# **WEB API-USSD**

**GATESMS.EU**  
wersja 0.9

## Spis

Historia wersji dokumentu.....	3
Bezpieczeństwo.....	3
Wymagania ogólne.....	3
Mechanizm zabezpieczenia transmisji HTTP.....	3
Zasady ogólne.....	3
Definicja usług zdalnych.....	4
Zasada działania komunikacji z serwerem USSD.....	4
Informacje dodatkowe o usłudze.....	4
Nawiązywanie połączenia z serwerem USSD - rozpoczęcie sesji.....	4
Przykład żądania – parametr body.....	4
Przykład odpowiedzi serwera.....	4
Komunikacja z aplikacją klienta.....	5
Nagłówki.....	5
Przykłady.....	5
Klasa obsługująca Web USSD Api.....	6

## Historia wersji dokumentu

- 0.9 - wersja robocza 25.01.2012

## Bezpieczeństwo

### Wymagania ogólne

1. Komunikacja w obu kierunkach musi uwzględniać zabezpieczenia transmisji HTTP opisane poniżej.

### Mechanizm zabezpieczenia transmisji HTTP

Zabezpieczenia dla usług zdalnych dostępnych przez HTTP są niezależne od metody i formatu przesyłanych wiadomości

### Zasady ogólne

Metody zdalne zwracają status http **mniejszy niż 300** w przypadku, gdy operacja się powiedzie. Zwykle będą to statusy **200** lub **204**. W przypadku, gdy operacja się nie powiedzie, usługa zwraca status błędu **większy bądź równy 400**. Dla uściślenia statusy od 400 włącznie do 500 wyłącznie oznaczają błąd w żądaniu (np. złe parametry - 400, błąd uwierzytelniania - 403, zły adres - 404). Statusy powyżej 500 włącznie oznaczają błąd po stronie serwera. Możliwe, że to samo żądanie po usunięciu usterki może zostać obsłużone poprawnie. Oznacza to, że dla statusów błędów poniżej 500 nie ma sensu ponawiać żądania, natomiast dla statusów od 500 włącznie, żądanie można ponawiać.

Cele mechanizmu bezpieczeństwa:

1. Uwierzytelnienie klienta i serwera
2. Weryfikacja poprawności i prawdziwości żądania i odpowiedzi

Serwer dla każdego klienta przygotowuje parę **KeyID** i **SecretKey**, gdzie **KeyID** jest identyfikatorem (*jawnym, login podany w procesie rejestracji*) a **SecretKey** jest kluczem ukrytym (*hasło, 10 znaków*). Aktualny tajny klucz można odczytać lub wygenerować po zalogowaniu w zakładce **DANE UŻYTKOWNIKA**. Obie te wartości są znane dla klienta i serwera.

Uwierzytelnianie opiera się o dwa nagłówki HTTP dodawane do żądania i odpowiedzi.

1. **X-GT-Timestamp** - timestamp operacji (unix timestamp)
2. **X-GT-Auth** - dodatkowy nagłówek do uwierzytelniania.

Przykład: X-GT-Auth: [KeyID]:[Hash] gdzie Hash to przygotowany skrót wiadomości opisany poniżej.

X-GT-Auth: 1234567:098f6bcd4621d373cade4e832627b4f6.

### Dla żądania

Hash = MD5(HTTP\_method + URI\_Path\_Query + MD5\_Body + Accept + X-GT-Timestamp + SecretKey)

Gdzie MD5 to funkcja haszująca a '+' oznacza konkatencję łańcuchów znaków oraz:

1. **HTTP\_method** to GET, POST, PUT itp.
2. **URI\_Path\_Query** to ścieżka i query string czyli część URL'a od pierwszego znaku '/' np: /ussd\_api.php
3. **MD5\_Body** - to MD5 z treści żądania o ile jest
4. **Accept** - nagłówek określający format odpowiedzi (zgodny ze standardem), przyjmuje wartości np. application/xml
5. **X-GT-Timestamp** - timestamp wykonania operacji (taki jak w nagłówku)
6. **SecretKey** - klucz klienta

### Przykład:

MD5(POST/ussd\_api.phpZXQW345YULAccept:application/xml12844675753LQBwxTiDnv)

Strona odbierająca żądanie ma obowiązek sprawdzenia, czy **Hash** jest poprawny. Jeśli nie jest to powinien zostać zwrócony status 403.

**Dla odpowiedzi**

Hash = MD5(MD5\_Body + Content-Type + X-GT-Timestamp + SecretKey)

gdzie **SecretKey** to klucz klienta, od którego otrzymaliśmy wiadomość

**Przykład**

```
MD5(9843f33LQd8xTiDnvGAKoc8n7pQ5qi3CW9SG584ContentType:application/xml12844675753LQBwxTiDnv)
```

Klient otrzymujący odpowiedź od serwera ma obowiązek sprawdzenia, czy odpowiedź jest podpisana poprawnie o ile żądanie zostało poprawnie obsłużone. Odpowiedzi o statusach  $\geq 400$  nie muszą być podpisane.

W przypadku błędu komunikaty błędów mogą być zamieszczone w odpowiedzi HTTP jako tekst.

**Definicja usług zdalnych**

Do transmisji danych wykorzystywany będzie głównie format XML. Obie strony mają obowiązek zapewnić zgodność przesyłanych dokumentów ze standardem XML. Dotyczy to w szczególności zasad używania znaków specjalnych w wiadomościach. Dokument XML przesyłany jest metodą POST.

Nagłówki Accept i Content-Type muszą być poprawnie ustawione, np.:

Content-Type: application/xml

Accept: application/xml

## Zasada działania komunikacji z serwerem USSD

Zainicjowanie sesji z usługą leży po stronie klienta. Klient nawiązując połączenie z serwerem przekazuje podstawowe parametry potrzebne do zainicjowania sesji. Po zainicjowaniu sesji serwer USSD komunikuje się z usługą zdalną za pomocą protokołu HTTP z żądaniem podania listy instrukcji(menu). Usługa zdalna odpowiada a serwer USSD przekazuje informację pod wskazany numer MSISDN. Cała sesja trwa aż do zakończenia przez usługę inicjującą lub przez usługę zdalną.

**Informacje dodatkowe o usłudze**

Długość wiadomości przy kodowaniu GSM7 wynosi 182 znaki a wiadomości kodowane w UCS2 mają maksymalną długość 66 znaków.

**Nawiązywanie połączenia z serwerem USSD - rozpoczęcie sesji**

Adres usługi : [http://www.gatesms.eu/ussd\\_api.php](http://www.gatesms.eu/ussd_api.php)

Metoda : POST

Content-Type: application/xml

Accept: application/xml

**Przykład żądania – parametr body**

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<session prefix="index1" msisdn="48505165092" language="GSM-7" >START</session>
```

**Przykład odpowiedzi serwera**

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><session>START OK</session>
```

**Element session:**

- Atrybuty obowiązkowe
  - prefix - unikalny identyfikator usługi klienta
  - msisdn - numer komórkowy z którym będzie prowadzona korespondencja
  - language - kodowanie, może przyjąć wartości GSM-7 lub UCS-2

**Komunikacja z aplikacją klienta**

Po poprawnym nawiązaniu sesji z serwerem USSD, serwer nawiązuje połączenie z aplikacją kliencką wysyłając do niej żądania podania instrukcji (menu). Definicja adresu zdalnej usługi konfigurowana jest po zalogowaniu do systemu w zakładce SMS -> USSD. Należy podać prefix oraz adres do usługi. Serwer USSD komunikuje się z usługą zdalną (klient) wysyłając nagłówki oraz treść dokumentu xml.

Metoda: POST

Content-Type: application/x-www-form-urlencoded

Accept: application/xml

**Nagłówki****X-GT-Session:** (String) - identyfikator sesji**X-GT-Action:** (String) start - wysyłany przy inicjowaniu sesji, wysyłany jako pierwszy komunikat | reponse - wysyłany w trakcie trwania sesji | stop - wysyłany do klienta aby zakończył sesję.**Przykłady**Przykład żądania – parametr body serwer ussd

Nagłówek: X-GT-Action: start

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><session msisdn="48787646535">abcd1234</session>
```

Przykład odpowiedzi klienta – parametr body

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<ussdmenu shouldClose="false" msisdn="48787646535">Dobry program\n1 - TAK\n2 - NIE</ussdmenu>
```

**Element ussdmenu:**

- Atrybuty obowiązkowe
  - shouldClose - atrybut informuje czy nadawany komunikat jest ostatnim i transmisja powinna zostać zakończona
  - msisdn - numer komórkowy

Przykład żądania – parametr body serwer ussd

Nagłówek: X-GT-Action: status

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><session msisdn="48787646535">abcd1234</session>
```

Przykład odpowiedzi klienta – parametr body

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><session isActive="true" msisdn="48787646535">abcd1234</session>
```

**Element session:**

- Atrybuty obowiązkowe
  - isActive - atrybut informuje czy aktualna sesja nadal jest utrzymywana przez aplikację klienta
  - msisdn - numer komórkowy

Przykład żądania – parametr body serwer ussd

Nagłówek: X-GT-Action: response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <session msisdn="48787646535"><text>1</text></session>
```

Przykład odpowiedzi klienta – parametr body

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<ussdmenu shouldClose="true" msisdn="48787646535">Do zobaczenia.</ussdmenu>
```

## Przykład żądania – parametr body serwer ussd

Nagłówek: X-GT-Action: stop

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><session>END</session>
```

## Przykład odpowiedzi klienta – parametr body

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><session>END OK</session>
```

## Dodatek A

### Klasa obsługująca Web USSD Api

Przykład obsługi klasy USSD\_CLIENT z pliku ussd\_client.php

Przykładowy plik workflow.php

Struktura plików źródłowych

ussd/

- >session/
  - >Session.php
- >data/
  - >http\_request.php
  - >sms\_xmlsender.php
  - > ussd\_client.php
- >workflow/
  - >workflow1.php
- >client/
  - >trigger.php

```
<?
include '../session/Session.php';
include '../data/ussd_client.php';
include '../data/sms_xmlsender.php';
include '../data/http_request.php';

//REQUEST
$http_request = new http_request();

//SESJA
$sessionID=$http_request->headers["X-GT-SESSION"]; //ID SESJI
$session = new Session($sessionID);
$Workflow=$session->getWorkflow( $sessionID );

if($Workflow) {
    $ussd=$Workflow;
} else {
    //Konstruktor klasy, oraz parametry
    $ussd = new USSD_CLIENT;
    $ussd->login="48603172099";
    $ussd->pass="0C27Ab3d03";
    $ussd->debug=0;
};

//LOGOWANIE
$Hash_serv=$ussd->serverLogin($http_request);

//ZADANIA
$xml=$ussd->parserXML($http_request->body);
//-----
if($http_request->headers["X-GT-ACTION"]=='status') {
    $Workflow=$session->getWorkflow( $sessionID );
    $isActive = ($Workflow) ? 'true' : 'false';
    $out=$ussd->getStats($isActive,$sessionID);
}
}
```

```
if($http_request->headers["X-GT-ACTION"]=='start') {
    $ussd->shouldClose='false';
    $ussd->requestCount=1;
    $ussd->menu="Twoje pierwsze menu testowe.\n1 - Zakoncz";
    $out=$ussd->getMenu();
    $session->saveSession( $sessionID, $ussd );
};

if($http_request->headers["X-GT-ACTION"]=='response') {
    if($ussd->text=='1' && $ussd->requestCount==1) {
        $ussd->shouldClose='true';
        $ussd->requestCount++;
        $ussd->menu="Usługa działa poprawnie";
        $out=$ussd->getMenu();
    }
    $session->saveSession( $sessionID, $ussd );
};

if($http_request->headers["X-GT-ACTION"]=='stop') {
    $out=$ussd->closeSession();
    $session->removeWorkflow( $sessionID );
};

$ussd->toOut($out);
exit;
?>
```