

elektroniczna Platforma Usług Administracji Publicznej

Instrukcja integracji z ePUAP w zakresie
interfejsów Profilu Zaufanego

wersja 02-02



INNOWACYJNA GOSPODARKA
NARODOWA STRATEGIA SPÓJNOŚCI



UNIA DLA PRZEDSIĘBIORCZYCH
PROGRAM KONKURENCYJNOŚĆ

UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ
ROZWOJU REGIONALNEGO



1.	WSTĘP	3
1.1.	CEL DOKUMENTU	3
1.2.	DOSTĘP DO ŚRODOWISKA TESTOWEGO.....	3
1.3.	SŁOWNIK I DODATKOWA DOKUMENTACJA	3
1.4.	WYMAGANE CERTYFIKATY	5
1.5.	PRAWA DOSTĘPU – ROLE.....	6
2.	OPIS FUNKCJONALNY INTERFEJSÓW PZ.....	7
2.1.	DOSTARCZANIE INFORMACJI O POSIADANIU PROFILU PZ	7
2.2.	PODPISYWANIE Z UŻYCIEM PZ	7
2.3.	WERYFIKACJA PODPISU PZ	8
3.	SPECYFIKACJA TECHNICZNA W POSTACI JAVADOC I WSDL	9
3.1.	JAVADOC.....	9
3.1.1.	hasTrustedProfileInstitution	10
3.1.2.	hasTrustedProfilePerson	11
3.1.3.	addDocumentToSigning	11
3.1.4.	getSignedDocument	12
3.1.5.	verifySignature	12
3.2.	STRUKTURA WYNIKU WERYFIKACJI PODPISU.....	13
3.3.	DEFINICJA WSDL	20
3.4.	ADRESY USŁUG	25

1. Wstęp

1.1. Cel dokumentu

Niniejsza instrukcja ma za zadanie stanowić pomoc dla użytkowników integrujących się z ePUAP w zakresie interfejsów podsystemu Profil Zaufany.

1.2. Dostęp do środowiska testowego

Środowisko testowe ePUAP jest dostępne pod adresem <http://test.epuap.gov.pl/wps/portal/epuap> . Instytucja, która planuje przeprowadzenie integracji wysyła zgłoszenie e-mail do MSWiA na adres: konrad.kaczmarek@mswia.gov.pl z określeniem nazwy Instytucji, danych kontaktowych, opisem Systemu oraz planowanym zakresem integracji. W zgłoszeniu należy również podać dane założonego konta na platformie testowej:

- Imię i Nazwisko administratora konta
 - Identyfikator użytkownika
 - Nazwę podmiotu
 - Identyfikator podmiotu
-

1.3. Słownik i dodatkowa dokumentacja

Pojęcie	Opis
System ePUAP	Ogół elementów wchodzących w skład platformy ePUAP.

Podsystem ePUAP	Część systemu ePUAP wyodrębniony w architekturze.
Interfejs	Definicja zbioru operacji, które mogą być wywoływane przez dwa komunikujące się elementy.
Komponent	Element składowy systemu (podsystemu) realizujący pewien spójny zakres funkcjonalności. Komunikacja między komponentami odbywa się wyłącznie przy użyciu interfejsów.
WebSerwis	Jedna z form implementacji interfejsu. Cechą interfejsu jest jego dostępność – może być wywołany zdalnie przez system zewnętrzny.
Profil Zaufany	Potwierdzony podpisem kwalifikowanym zbiór informacji wiążący profil ePUAP i zweryfikowane dane.
Aplikacja PZ (Profil Zaufany)	Aplikacja dostarczająca funkcjonalności obsługi i zarządzania profilami zaufanymi
Podpis PZ (podpis profilem zaufanym)	Podpis cyfrowy dokonywany przez ePUAP zawierający informację o podmiocie zaufanym w imieniu, którego dokonano podpisu. Dodatkowe informacje umieszczone są w elemencie claimedRole struktury podpisu.
HSM	Hardware Security Module – sprzętowy moduł kryptograficzny
TSA	Time Stamping Authority – usługi służące do

	oznaczania czasem
SOPEL	System Obsługi Podpisu Elektronicznego
Konsola DRACO	Konsola zarządzania uprawnieniami w ePUAP dostępna pod adresem https://konsoladraco.epuap.gov.pl/DracoConsole

Odnośniki:

- Pomoc dla integratorów ePUAP: materiały znajdują się na portalu ePUAP na stronach: pomoc->Pomoc dla integratorów
- Instrukcja Podsystem bezpieczeństwa – plik dostępny na stronach ePUAP: pomoc-> Instrukcje Administratora Podmiotu (Plik dostępny tylko dla zalogowanych użytkowników z uprawnieniami instytucji publicznej.)
- PZ_Instrukcja_OP.doc – instrukcja użytkownika dla Osoby Potwierdzającej
- PZ_Instrukcja_uzytkownika_koncowego – instrukcja użytkownika końcowego (korzystającego z podpisywania za pomocą PZ)

1.4. Wymagane certyfikaty

W komunikacji pomiędzy ePUAP, a systemami zewnętrznymi, stosowany jest standardowy mechanizm WS-Security. Usługi sieciowe ePUAP wymagają, aby element `soap:body` przesyłanej wiadomości, stanowiącej wywołania operacji, podpisany był certyfikatem zarejestrowanym w systemie DRACO (proces rejestracji opisano w dokumentacji dodatkowej – rozdział 1.3.). W przychodzących wiadomościach weryfikowana jest obecność tokenu zgodnego z

<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3>

Wiadomości wychodzące z ePUAP oraz odpowiedzi wywołań operacji web serwisów autoryzowane są w analogiczny sposób – poprzez podpisanie elementu `soap:body` certyfikatem ePUAP (certyfikaty są dostępne na stronach portalu ePUAP: epuap.gov.pl -> Pomoc -> Pomoc dla integratorów -> Dokumentacja i przykłady -> Specyfikacja interfejsów WSDL). System zewnętrzny odbierający wiadomości pochodzące z systemu ePUAP powinien weryfikować podpis ePUAP na tych wiadomościach.

W systemie ePUAP, zarówno na środowisku produkcyjnym jak i testowym, do zarejestrowania systemu zewnętrznego w konsoli Draco wymagany jest certyfikat niekwalifikowany serwerowy z rozszerzeniem "uwierzytelnianie klienta". Lista ogólnodostępnych wystawców certyfikatów niekwalifikowanych serwerowych z rozszerzeniem "uwierzytelnianie klienta":

SIGILLUM: Sigillum PCCE - CA;

UNIZETO: Certum Level II CA (W ofercie Unizeto certyfikat ten nazywa się Commercial SSL);

KIR: SZAFIR 31 CA (należy jawnie zgłosić potrzebę rozszerzenia "Uwierzytelnianie klienta").

1.5. Prawa dostępu – role

Usługi aplikacji Profilu Zaufanego wymagają zarejestrowania systemu zewnętrznego w DRACO (opis tego procesu dostępny jest w dokumentacji dodatkowej rozdział 1.3.). Zarejestrowane systemy mają dostęp do wszystkich usług udostępnianych przez WebServices (patrz rozdział 2.)

2. Opis funkcjonalny interfejsów PZ

2.1. Dostarczanie informacji o posiadaniu profilu PZ

Interfejs dostarcza informacji o posiadaniu profilu zaufanego i odpowiada rezultatem logicznym (prawda, fałsz) po wywołaniu operacji „hasTrustedProfileInstitution” oraz „hasTrustedProfilePerson” odpowiednio dla PZ podmiotu i PZ użytkownika. Szczegóły w sekcji [JavaDoc](#)

2.2. Podpisywanie z użyciem PZ

Podpisywanie przez systemy zewnętrzne opisuje poniższy przebieg:

1. System zewnętrzny za pośrednictwem usługi WebServices (WSSecurity) rejestruje zlecenie podpisu dokumentu. W rezultacie otrzymuje URL, pod który ma przekierować użytkownika.
2. Użytkownik jest przekierowywany przez System zewnętrzny na adres otrzymany w pkt.1. ePUAP weryfikuje czy użytkownik jest zalogowany na ePUAP (również przez SSO) i w razie potrzeby automatycznie kieruje go na stronę logowania. Następnie użytkownik podpisuje dokument. Jeśli operacja się powiedzie ePUAP przekierowuje użytkownika na stronę przekazaną w zleceniu (pkt.1) w parametrze "successURL", w przeciwnym wypadku ePUAP przekierowuje użytkownika na stronę określoną w parametrze "failureURL" (w zleceniu pkt.1).
3. System zewnętrzny za pośrednictwem usługi WebServices (WSSecurity) pobiera podpisany dokument z ePUAP. Po udanym pobraniu pliku jest on usuwany z bazy. W przypadku wystąpienia błędu plik pozostanie w bazie w oczekiwaniu na kolejną próbę pobrania a klient WebService’u zostanie poinformowany o przyczynie błędu (zostanie rzucony wyjątek).

Szczegóły w sekcji [JavaDoc](#)

2.3. Weryfikacja podpisu PZ

Operacja „verifySignature” umożliwia przekazanie dokumentu podpisanego przy pomocy PZ, który będzie poddany weryfikacji. Rezultatem weryfikacji jest struktura XML zawierająca wszystkie niezbędne informacje dotyczące podpisu dokumentu. Operacja weryfikacji obsługuje zarówno podpis PZ jak i podpis certyfikatem. Wynik będzie zawierał informacje opisujące oba typy podpisu. Szczegóły w sekcji [JavaDoc](#)

3. Specyfikacja techniczna w postaci javadoc i WSDL

3.1. JavaDoc

Interface ITPSigning

All Superinterfaces: java.rmi.Remote

```
public interface ITPSigning
extends java.rmi.Remote
```

WebService udostępniający usługi związane z podpisywaniem Profilem Zaufanym

Method Summary

Zwracany typ	Sygnatura funkcji
java.lang.String	<u>addDocumentToSigning</u> (byte[] doc, java.lang.String successURL, java.lang.String failureURL, java.lang.String additionalInfo) Umożliwia dodanie dokumentu do podpisania Profilem Zaufanym.
byte[]	<u>getSignedDocument</u> (java.lang.String id) Zwraca podpisany dokument przekazany w wywołaniu funkcji <u>addDocumentToSigning</u> (byte[], <u>String</u> , <u>String</u> , <u>String</u>)
boolean	<u>hasTrustedProfileInstitution</u> (java.lang.Stri

	ng tgsid) Zwraca informację czy dla podanego tgsid istnieje Profil Zaufany instytucji
boolean	hasTrustedProfilePerson (java.lang.String tgsid) Zwraca informację czy dla podanego tgsid istnieje Profil Zaufany użytkownika
java.lang.String	verifySignature (byte[] document) Zwraca status weryfikacji podpisu pod dokumentem

Method Detail

3.1.1. hasTrustedProfileInstitution

boolean **hasTrustedProfileInstitution**(java.lang.String tgsid)
throws
gov.comarch.ws.exception.WSSigningException

Zwraca informację czy dla podanego tgsid istnieje Profil Zaufany instytucji

Parameters:

tgsid – token bezpieczeństwa nadawany przez DRACO w momencie logowania użytkownika do aplikacji ePUAP. Token (TGSID) jest ciągiem tekstowym przekazywanym w formie Cookies.

Returns:

true jeżeli istnieje Profil Zaufany organizacji dla podanego tgsig i użytkownik ma uprawnienia do jego używania

Throws:

gov.comarch.ws.exception.WSSigningException - wyjątek zawierający przyczynę błędu (np. niepoprawne parametry wejściowe, błąd komunikacji z modułem PZ, błąd komunikacji z bazą itp.)

3.1.2. hasTrustedProfilePerson

```
boolean hasTrustedProfilePerson(java.lang.String tgsid)
    throws
gov.comarch.ws.exception.WSSigningException
```

Zwraca informację czy dla podanego tgsid istnieje Profil Zaufany użytkownika

Parameters:

`tgsid` - token bezpieczeństwa nadawany przez DRACO w momencie logowania użytkownika do aplikacji ePUAP. Tgsid aktualnie zalogowanego użytkownika można pobrać z ciasteczka o nazwie TGSID.

Returns:

true, jeżeli istnieje Profil Zaufany użytkownika

Throws:

`gov.comarch.ws.exception.WSSigningException` - wyjątek zawierający przyczynę błędu (np. niepoprawne parametry wejściowe, błąd komunikacji z modułem PZ, błąd komunikacji z bazą itp.)

3.1.3. addDocumentToSigning

```
java.lang.String addDocumentToSigning(byte[] doc,
    java.lang.String successURL,
    java.lang.String failureURL,
    java.lang.String additionalInfo)
    throws
gov.comarch.ws.exception.WSSigningException
```

Umożliwia dodanie dokumentu do podpisania Profilem Zaufanym.

Parameters:

`doc` - dokument do podpisania , rozmiar wejściowego dokumentu nie może być większy niż 5 MB.

`successURL` - url, na który zostanie przekierowany użytkownik w przypadku udanej próby podpisania dokumentu

`failureURL` - url, na który zostanie przekierowany użytkownik w przypadku nieudanej próby podpisania dokumentu

`additionalInfo` - informacje dodatkowe w postaci tekstu prezentowanego na stronie, na której użytkownik podpisuje przekazany dokument

Returns:

URL do strony umożliwiającej podpisanie przekazanego dokumentu

Throws:

`gov.comarch.ws.exception.WSSigningException` - wyjątek zawierający informacje o przyczynienie błędu (np. niepoprawne parametry wejściowe w tym za duży rozmiar pliku lub plik pusty, błąd komunikacji z modułem PZ, błąd komunikacji z bazą itp.)

3.1.4. getSignedDocument

`byte[] getSignedDocument(java.lang.String id)`
throws `gov.comarch.ws.exception.WSSigningException`

Zwraca podpisany dokument przekazany w wywołaniu funkcji [addDocumentToSigning\(byte\[\], String, String, String\)](#)

Parameters:

`id` - klucz identyfikujący podpisany dokument (adres zwrócony przez funkcję [addDocumentToSigning\(byte\[\], String, String, String\)](#))

Returns:

podpisany dokument

Throws:

`gov.comarch.ws.exception.WSSigningException` – wyjątek zawierający informacje o przyczynie błędu (np. niepoprawne parametry wejściowe, nie istnieje podpisany dokument dla podanego id itp.)

3.1.5. verifySignature

`java.lang.String verifySignature(byte[] document)`
throws `gov.comarch.ws.exception.WSSigningException`

Zwraca status weryfikacji podpisu pod dokumentem. Status weryfikacji jest w formie XML i zawiera informacje o każdym podpisie pod dokumentem (w przypadku gdy jest to podpis wielokrotny). Jeżeli dany podpis będzie podpisem

Profilom Zaufanym zostanie umieszczona odpowiednia informacja. Dokładne informacje o pliku weryfikacji podano niżej DODAĆ

Parameters:

`document` - podpisany dokument , rozmiar dokumentu nie może być większy niż 5,5 MB.

Returns:

status weryfikacji w formie XML (strukturę opisano w pkt. 3.2.)

Throws:

`gov.comarch.ws.exception.WSSigningException` - wyjątek zawierający informacje o przyczynie błędu

3.2. Struktura wyniku weryfikacji podpisu

VerifyResult – korzeń dokumentu

- **ValidDocumentSignature** – element typu boolean, zawierający informację czy dokument jest poprawnie podpisany. Atrybut „znaczenie” jest opisową zawartością tego pola (Prawidłowy, Nieprawidłowy).
- **SignatureType** – zawiera typ podpisu (XAdES, PAdES, CAdES)
- **GenerationTime** – data i godzina wygenerowania tego dokumentu xml
- **StatusInfo** – dla każdego podpisu w dokumencie tworzona jest oddzielna struktura, której korzeniem jest ten właśnie element, w przypadku braku podpisu w dokumencie również tworzona jest jedna tak struktura.
 - ValidSignature - element typu boolean, zawierający informację czy podpis jest poprawny. Atrybut „znaczenie” jest opisową zawartością tego pola (Prawidłowy, Nieprawidłowy).
 - VerifyStatus – element typu int ze statusem weryfikacji podpisu. Atrybut „znaczenie” jest opisową zawartością tego pola.
 - VerifySignerCert - element typu int z informacją o certyfikacie użytym w podpisie. Atrybut „znaczenie” jest opisową zawartością tego pola.

- VerifySignerCertUsage - element typu int z informacją o sposobie użycia certyfikatu wykorzystanego w podpisie. Atrybut „znaczenie” jest opisową zawartością tego pola, a dodatkowo w przypadku gdy są prawdziwe ustawiane są atrybuty „kwalifikowany”, „logowanie”, „ocspZaufane”, „ocspOdpowiedz”, „ocspPrzekierowanie”, „upo”, „epo” oraz „tsa”.
- CommitmentType – wartość pola „Commitment type” z podpisu.
- GracePeriod - wartość pola „Grace period” z podpisu.
- ParentSignatureId – ID podpisu nadrzędnego (używane przy kontrasygnacie).
- SignatureCertIssuer – zawiera informację o danych wystawcy certyfikatu, a w atrybutach poszczególne pola z tej informacji.
- SignatureCertSerial – numer seryjny certyfikatu.
- SignatureCertSubject – podobnie jak SignatureCertIssuer, ale zawiera informacje o właścicielu certyfikatu.
- SignatureId – ID podpisu.
- SigningTime – data i godzina podpisania dokumentu.
- UriID – wskaźniki do podpisanych elementów.
- SignatureTimeStamp – informacje o oznaczeniu czasem podpisu, w przypadku braku oznaczenia czasem atrybut „znaczenie” przyjmuje wartość „Brak oznaczenia czasem”.
 - TimeStampTime – czas oznaczenia podpisu.
 - VerifyStatus - element typu int ze statusem oznaczenia czasem. Atrybut „znaczenie” jest opisową zawartością tego pola.
- ArchiveTimeStamp - informacje o postaci archiwalnej podpisu, w przypadku braku atrybut „znaczenie” przyjmuje wartość „Brak postaci archiwalnej”.
 - TimeStampTime – czas oznaczenia podpisu.
 - VerifyStatus - element typu int ze statusem oznaczenia czasem. Atrybut „znaczenie” jest opisową zawartością tego pola.

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
- <xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
```

```
- <xs:element name="VerifyResult">
```

```
- <xs:complexType>
```

```

- <xs:sequence>
- <xs:element name="ValidDocumentSignature">
- <xs:complexType>
- <xs:simpleContent>
- <xs:extension base="xs:boolean">
- <xs:attribute name="znaczenie" use="required">
- <xs:simpleType>
- <xs:restriction base="xs:string">
  <xs:pattern value="Prawidłowy|Nieprawidłowy" />
  </xs:restriction>
  </xs:simpleType>
  </xs:attribute>
  </xs:extension>
  </xs:simpleContent>
  </xs:complexType>
  </xs:element>
  <xs:element name="SignatureType" type="xs:string" />
  <xs:element name="GenerationTime" type="xs:string" />
- <xs:element maxOccurs="unbounded" name="StatusInfo">
- <xs:complexType>
- <xs:sequence>
- <xs:element name="ValidSignature">
- <xs:complexType>
- <xs:simpleContent>
- <xs:extension base="xs:boolean">
- <xs:attribute name="znaczenie" use="required">
- <xs:simpleType>
- <xs:restriction base="xs:string">
  <xs:pattern value="Prawidłowy|Nieprawidłowy" />
  </xs:restriction>
  </xs:simpleType>
  </xs:attribute>
  </xs:extension>
  </xs:simpleContent>
  </xs:complexType>
  </xs:element>
- <xs:element name="VerifyStatus">

```

```

- <xs:complexType>
- <xs:simpleContent>
- <xs:extension base="xs:int">
  <xs:attribute name="znaczenie" type="xs:string" use="required" />
  <xs:attribute name="komunikat" type="xs:string" use="optional" />
  </xs:extension>
  </xs:simpleContent>
  </xs:complexType>
  </xs:element>
- <xs:element name="VerifySignerCert">
- <xs:complexType>
- <xs:simpleContent>
- <xs:extension base="xs:int">
- <xs:attribute name="znaczenie" use="required">
- <xs:simpleType>
- <xs:restriction base="xs:string">
  <xs:pattern value="\[brak\]|Ważny|Nieważny|Unieważniony|Nieznany
  wystawca|Brak OCSP lub CRL|Błędny|(Nieznany błąd [0-9]+)" />
  </xs:restriction>
  </xs:simpleType>
  </xs:attribute>
  </xs:extension>
  </xs:simpleContent>
  </xs:complexType>
  </xs:element>
- <xs:element name="VerifySignerCertUsage">
- <xs:complexType>
- <xs:simpleContent>
- <xs:extension base="xs:int">
  <xs:attribute name="znaczenie" type="xs:string" use="required" />
  <xs:attribute name="kwalifikowany" type="xs:boolean" use="optional" />
  <xs:attribute name="logowanie" type="xs:boolean" use="optional" />
  <xs:attribute name="ocspZaufane" type="xs:boolean" use="optional" />
  <xs:attribute name="ocspOdpowiedz" type="xs:boolean" use="optional" />
  <xs:attribute name="ocspPrzekierowanie" type="xs:boolean" use="optional" />
  <xs:attribute name="upo" type="xs:boolean" use="optional" />
  <xs:attribute name="epo" type="xs:boolean" use="optional" />

```

```

<xs:attribute name="tsa" type="xs:boolean" use="optional" />
  </xs:extension>
  </xs:simpleContent>
  </xs:complexType>
  </xs:element>
  <xs:element name="CommitmentType" type="xs:string" />
  <xs:element name="GracePeriod" type="xs:int" />
  <xs:element name="ParentSignatureId" type="xs:string" />
- <xs:element name="SignatureCertIssuer">
- <xs:complexType>
- <xs:simpleContent>
- <xs:extension base="xs:string">
  <xs:anyAttribute />
  </xs:extension>
  </xs:simpleContent>
  </xs:complexType>
  </xs:element>
  <xs:element name="SignatureCertSerial" type="xs:string" />
- <xs:element name="SignatureCertSubject">
- <xs:complexType>
- <xs:simpleContent>
- <xs:extension base="xs:string">
  <xs:anyAttribute />
  </xs:extension>
  </xs:simpleContent>
  </xs:complexType>
  </xs:element>
  <xs:element name="SignatureId" type="xs:string" />
  <xs:element name="SigningTime" type="xs:string" />
- <xs:element minOccurs="0" maxOccurs="unbounded" name="UriID">
- <xs:complexType>
- <xs:simpleContent>
- <xs:extension base="xs:string">
  <xs:attribute name="lp" type="xs:int" use="required" />
  </xs:extension>
  </xs:simpleContent>
  </xs:complexType>

```

```

    </xs:element>
- <xs:element name="SignatureTimeStamp">
- <xs:complexType>
- <xs:sequence minOccurs="0">
    <xs:element name="TimeStampTime" type="xs:string" />
- <xs:element name="VerifyStatus">
- <xs:complexType>
- <xs:simpleContent>
- <xs:extension base="xs:int">
- <xs:attribute name="znaczenie" use="required">
- <xs:simpleType>
- <xs:restriction base="xs:string">
    <xs:pattern value="\[brak\]|Brak znacznika|Znacznik prawidłowy|Znacznik
nieprawidłowy|Nieważny certyfikat OCSP|Niezaufany certyfikat
OCSP|Nieważny certyfikat TSA|Niezaufany certyfikat TSA|(Nieznany status
[0-9]+)" />
    </xs:restriction>
    </xs:simpleType>
    </xs:attribute>
    </xs:extension>
    </xs:simpleContent>
    </xs:complexType>
    </xs:element>
    </xs:sequence>
- <xs:attribute name="znaczenie" use="optional">
- <xs:simpleType>
- <xs:restriction base="xs:string">
    <xs:pattern value="Brak oznaczenia czasem" />
    </xs:restriction>
    </xs:simpleType>
    </xs:attribute>
    </xs:complexType>
    </xs:element>
- <xs:element name="ArchiveTimeStamp">
- <xs:complexType>
- <xs:sequence minOccurs="0">
    <xs:element name="TimeStampTime" type="xs:string" />
- <xs:element name="VerifyStatus">

```

```

- <xs:complexType>
- <xs:simpleContent>
- <xs:extension base="xs:int">
- <xs:attribute name="znaczenie" use="required">
- <xs:simpleType>
- <xs:restriction base="xs:string">
  <xs:pattern value="\[brak\]|Brak znacznika|Znacznik prawidłowy|Znacznik
nieprawidłowy|Nieważny certyfikat OCSP|Niezaufany certyfikat
OCSP|Nieważny certyfikat TSA|Niezaufany certyfikat TSA|(Nieznany status
[0-9]+)" />
</xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
</xs:element>
</xs:sequence>
- <xs:attribute name="znaczenie" use="optional">
- <xs:simpleType>
- <xs:restriction base="xs:string">
  <xs:pattern value="Brak postaci archiwalnej" />
  </xs:restriction>
  </xs:simpleType>
  </xs:attribute>
  </xs:complexType>
  </xs:element>
- <xs:element name="ZP">
- <xs:complexType>
- <xs:sequence minOccurs="0">
  <xs:any />
  </xs:sequence>
  <xs:attribute name="czy_obecny" type="xs:boolean" use="required" />
  </xs:complexType>
  </xs:element>
  </xs:sequence>
  </xs:complexType>
  </xs:element>

```

```

</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>

```

3.3. Definicja WSDL

```

<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://signing.ws.comarch.gov"
xmlns:impl="http://signing.ws.comarch.gov"
xmlns:intf="http://signing.ws.comarch.gov"
xmlns:tns2="http://exception.ws.comarch.gov"
xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/"
xmlns:wSDLsoap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsi="http://ws-
i.org/profiles/basic/1.1/xsd" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
    <schema targetNamespace="http://signing.ws.comarch.gov"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <element name="tgsid" nillable="true" type="xsd:string"/>
      <element name="hasTrustedProfileInstitutionReturn" type="xsd:boolean"/>
      <element name="string" nillable="true" type="xsd:string"/>
      <element name="hasTrustedProfilePersonReturn" type="xsd:boolean"/>
      <element name="doc" type="xsd:base64Binary"/>
      <element name="successURL" nillable="true" type="xsd:string"/>
      <element name="failureURL" nillable="true" type="xsd:string"/>
      <element name="additionalInfo" nillable="true" type="xsd:string"/>
      <element name="addDocumentToSigningReturn" nillable="true" type="xsd:string"/>
      <element name="id" nillable="true" type="xsd:string"/>
      <element name="getSignedDocumentReturn" type="xsd:base64Binary"/>
      <element name="document" type="xsd:base64Binary"/>
      <element name="verifySignedDocumentReturn" nillable="true" type="xsd:string"/>
    </schema>
    <schema targetNamespace="http://exception.ws.comarch.gov"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <complexType name="WSSigningException">
        <sequence>
          <element name="code" nillable="true" type="xsd:string"/>
          <element name="errorMessage" nillable="true" type="xsd:string"/>
        </sequence>
      </complexType>
      <element name="WSSigningException" nillable="true"
type="tns2:WSSigningException"/>
    </schema>
  </wsdl:types>

  <wsdl:message name="verifySignedDocumentRequest">
    <wsdl:part name="document" type="xsd:base64Binary"/>
  </wsdl:message>

  <wsdl:message name="addDocumentToSigningResponse">

```

```

    <wsdl:part name="addDocumentToSigningReturn" type="xsd:string"/>
</wsdl:message>

<wsdl:message name="getSignedDocumentResponse">
    <wsdl:part name="getSignedDocumentReturn" type="xsd:base64Binary"/>
</wsdl:message>

<wsdl:message name="hasTrustedProfilePersonRequest">
    <wsdl:part name="tgsid" type="xsd:string"/>
</wsdl:message>

<wsdl:message name="hasTrustedProfileInstitutionResponse">
    <wsdl:part name="hasTrustedProfileInstitutionReturn" type="xsd:boolean"/>
</wsdl:message>

<wsdl:message name="WSSigningException">
    <wsdl:part name="fault" type="tns2:WSSigningException"/>
</wsdl:message>

<wsdl:message name="hasTrustedProfileInstitutionRequest">
    <wsdl:part name="tgsid" type="xsd:string"/>
</wsdl:message>

<wsdl:message name="getSignedDocumentRequest">
    <wsdl:part name="id" type="xsd:string"/>
</wsdl:message>

<wsdl:message name="hasTrustedProfilePersonResponse">
    <wsdl:part name="hasTrustedProfilePersonReturn" type="xsd:boolean"/>
</wsdl:message>

<wsdl:message name="verifySignedDocumentResponse">
    <wsdl:part name="verifySignedDocumentReturn" type="xsd:string"/>
</wsdl:message>

<wsdl:message name="addDocumentToSigningRequest">
    <wsdl:part name="doc" type="xsd:base64Binary"/>

    <wsdl:part name="successURL" type="xsd:string"/>

    <wsdl:part name="failureURL" type="xsd:string"/>

    <wsdl:part name="additionalInfo" type="xsd:string"/>
</wsdl:message>

<wsdl:portType name="TPSigning">
    <wsdl:operation name="hasTrustedProfileInstitution" parameterOrder="tgsid">
        <wsdl:input message="intf:hasTrustedProfileInstitutionRequest"
name="hasTrustedProfileInstitutionRequest"/>

```

```

        <wsdl:output message="intf:hasTrustedProfileInstitutionResponse"
name="hasTrustedProfileInstitutionResponse"/>

        <wsdl:fault message="intf:WSSigningException" name="WSSigningException"/>
    </wsdl:operation>

    <wsdl:operation name="hasTrustedProfilePerson" parameterOrder="tgsid">
        <wsdl:input message="intf:hasTrustedProfilePersonRequest"
name="hasTrustedProfilePersonRequest"/>

        <wsdl:output message="intf:hasTrustedProfilePersonResponse"
name="hasTrustedProfilePersonResponse"/>

        <wsdl:fault message="intf:WSSigningException" name="WSSigningException"/>
    </wsdl:operation>

    <wsdl:operation name="addDocumentToSigning" parameterOrder="doc successURL
failureURL additionalInfo">
        <wsdl:input message="intf:addDocumentToSigningRequest"
name="addDocumentToSigningRequest"/>

        <wsdl:output message="intf:addDocumentToSigningResponse"
name="addDocumentToSigningResponse"/>

        <wsdl:fault message="intf:WSSigningException" name="WSSigningException"/>
    </wsdl:operation>

    <wsdl:operation name="getSignedDocument" parameterOrder="id">
        <wsdl:input message="intf:getSignedDocumentRequest"
name="getSignedDocumentRequest"/>

        <wsdl:output message="intf:getSignedDocumentResponse"
name="getSignedDocumentResponse"/>

        <wsdl:fault message="intf:WSSigningException" name="WSSigningException"/>
    </wsdl:operation>

    <wsdl:operation name="verifySignedDocument" parameterOrder="document">
        <wsdl:input message="intf:verifySignedDocumentRequest"
name="verifySignedDocumentRequest"/>

        <wsdl:output message="intf:verifySignedDocumentResponse"
name="verifySignedDocumentResponse"/>

        <wsdl:fault message="intf:WSSigningException" name="WSSigningException"/>
    </wsdl:operation>
</wsdl:portType>

<wsdl:binding name="TPSigningSoapBinding" type="intf:TPSigning">
<wsaw:UsingAddressing wsdl:required="false"
xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl"/>

    <wsdlsoap:binding style="rpc"
transport="http://schemas.xmlsoap.org/soap/http"/>

```

```
<wsdl:operation name="hasTrustedProfileInstitution">
  <wsdlsoap:operation soapAction="hasTrustedProfileInstitution"/>

  <wsdl:input name="hasTrustedProfileInstitutionRequest">
    <wsdlsoap:body namespace="http://signing.ws.comarch.gov" use="literal"/>
  </wsdl:input>

  <wsdl:output name="hasTrustedProfileInstitutionResponse">
    <wsdlsoap:body namespace="http://signing.ws.comarch.gov" use="literal"/>
  </wsdl:output>

  <wsdl:fault name="WSSigningException">
    <wsdlsoap:fault name="WSSigningException" use="literal"/>
  </wsdl:fault>
</wsdl:operation>

<wsdl:operation name="hasTrustedProfilePerson">
  <wsdlsoap:operation soapAction="hasTrustedProfilePerson"/>

  <wsdl:input name="hasTrustedProfilePersonRequest">
    <wsdlsoap:body namespace="http://signing.ws.comarch.gov" use="literal"/>
  </wsdl:input>

  <wsdl:output name="hasTrustedProfilePersonResponse">
    <wsdlsoap:body namespace="http://signing.ws.comarch.gov" use="literal"/>
  </wsdl:output>

  <wsdl:fault name="WSSigningException">
    <wsdlsoap:fault name="WSSigningException" use="literal"/>
  </wsdl:fault>
</wsdl:operation>

<wsdl:operation name="addDocumentToSigning">
  <wsdlsoap:operation soapAction="addDocumentToSigning"/>

  <wsdl:input name="addDocumentToSigningRequest">
    <wsdlsoap:body namespace="http://signing.ws.comarch.gov" use="literal"/>
  </wsdl:input>

  <wsdl:output name="addDocumentToSigningResponse">
    <wsdlsoap:body namespace="http://signing.ws.comarch.gov" use="literal"/>
  </wsdl:output>

  <wsdl:fault name="WSSigningException">
    <wsdlsoap:fault name="WSSigningException" use="literal"/>
  </wsdl:fault>
</wsdl:operation>

<wsdl:operation name="getSignedDocument">
```

```

<wsdlsoap:operation soapAction="getSignedDocument"/>

<wsdl:input name="getSignedDocumentRequest">
  <wsdlsoap:body namespace="http://signing.ws.comarch.gov" use="literal"/>
</wsdl:input>

<wsdl:output name="getSignedDocumentResponse">
  <wsdlsoap:body namespace="http://signing.ws.comarch.gov" use="literal"/>
</wsdl:output>

<wsdl:fault name="WSSigningException">
  <wsdlsoap:fault name="WSSigningException" use="literal"/>
</wsdl:fault>
</wsdl:operation>

<wsdl:operation name="verifySignedDocument">
  <wsdlsoap:operation soapAction="verifySignedDocument"/>

  <wsdl:input name="verifySignedDocumentRequest">
    <wsdlsoap:body namespace="http://signing.ws.comarch.gov" use="literal"/>
  </wsdl:input>

  <wsdl:output name="verifySignedDocumentResponse">
    <wsdlsoap:body namespace="http://signing.ws.comarch.gov" use="literal"/>
  </wsdl:output>

  <wsdl:fault name="WSSigningException">
    <wsdlsoap:fault name="WSSigningException" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
</wsdl:binding>

<wsdl:service name="TPSigningService">
  <wsdl:port binding="intf:TPSigningSoapBinding" name="TPSigning">
    <wsdlsoap:address
location="http://<<<adres_srodowiska>>>/zp_signing_external_ws/services/TPSigning
"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```

3.4. Adresy usług

Środowisko testowe

Usługa	Adres usługi web service
ITPSigning	https://test.epuap.gov.pl/zp_signing_external_ws/services/TPSigning

Środowisko produkcyjne

Usługa	Adres usługi web service
ITPSigning	https://ws.epuap.gov.pl/zp_signing_external_ws/services/TPSigning