

```

<?xml version="1.0" encoding="utf-8"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:conv="http://www.openuri.org/2002/04/soap/conversation/"
xmlns:cw="http://www.openuri.org/2002/04/wsdl/conversation/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:jms="http://www.openuri.org/2002/04/wsdl/jms/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:s0="http://fsb.belgium.be/prove"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
targetNamespace="http://fsb.belgium.be/prove">
  <types>
    <s:schema xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:fph="http://fsb.belgium.be/fphp/v2_1/fphp100"
xmlns:fph2="http://fsb.belgium.be/fphp/v2_1/fphp390"
elementFormDefault="qualified" targetNamespace="http://fsb.belgium.be/prove">
      <s:import namespace="http://fsb.belgium.be/fphp/v2_1/fphp100"/>
      <s:import namespace="http://fsb.belgium.be/fphp/v2_1/fphp390"/>
      <s:element name="fphp100">
        <s:complexType>
          <s:sequence>
            <s:element ref="fph:fphp100"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="fphp100Response">
        <s:complexType>
          <s:sequence>
            <s:element ref="fph2:fphp390"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="ping">
        <s:complexType>
          <s:sequence/>
        </s:complexType>
      </s:element>
      <s:element name="pingResponse">
        <s:complexType>
          <s:sequence>
            <s:element name="pingResult" type="s:string" minOccurs="0"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="string" nillable="true" type="s:string"/>
      <s:element name="testSOAPFault">
        <s:complexType>
          <s:sequence/>
        </s:complexType>
      </s:element>
      <s:element name="testSOAPFaultResponse">
        <s:complexType>
          <s:sequence/>
        </s:complexType>
      </s:element>
    </s:schema>
    <!-- edited with XMLSpy v2006 rel. 3 sp2 (http://www.altova.com) by Smals-
MvM vzw (Smals-MvM vzw) -->
    <xs:schema xmlns:iso="http://fsb.belgium.be/common/isocodes"
xmlns:per="http://fsb.belgium.be/prove/person"
xmlns:not="http://fsb.belgium.be/prove/notary"
xmlns="http://fsb.belgium.be/fphp/v2_1/fphp100"

```

```

xmlns:eid="http://fsb.belgium.be/common/eid"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://fsb.belgium.be/fphp/v2_1/fphp100"
elementFormDefault="qualified">
  <xs:annotation>
    <xs:documentation xml:lang="en">Schema version: 1.0
      Date: 2006-11-07
      Author: Quentin Dallons</xs:documentation>
  </xs:annotation>
  <xs:import namespace="http://fsb.belgium.be/prove/notary"/>
  <xs:import namespace="http://fsb.belgium.be/prove/person"/>
  <xs:import namespace="http://fsb.belgium.be/common/isocodes"/>
  <xs:import namespace="http://fsb.belgium.be/common/eid"/>
  <xs:simpleType name="t_SimpleString">
    <xs:annotation>
      <xs:documentation>basic string type that allows only alphabetic
characters, numbers, whitespace, and -</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="collapse"/>
      <xs:pattern value="[\p{L}\p{N}\s\-\]*/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="t_NISCode">
    <xs:annotation>
      <xs:documentation>base type for NIS codes</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="99999"/>
      <xs:whiteSpace value="collapse"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="GenderCodeType">
    <xs:annotation>
      <xs:documentation>The gender code (0 = unspecified, 1 = male, 2 =
female)</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:integer">
      <xs:whiteSpace value="collapse"/>
      <xs:enumeration value="0"/>
      <xs:enumeration value="1"/>
      <xs:enumeration value="2"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="t_IncompleteDate">
    <xs:annotation>
      <xs:documentation>a possibly incomplete date. Format yyyy-mm-dd,
yyyy-mm-00 or yyyy-00-00</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:length value="10"/>
      <xs:whiteSpace value="collapse"/>
      <xs:pattern value="[1-2][0-9]{3}\-[0-1][0-9]\-[0-3][0-9]"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="fphp100">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="client_application_id" type="xs:string"/>
        <xs:element ref="not:notary"/>
        <xs:element name="search_person">
          <xs:complexType>

```

```

<xs:sequence>
  <xs:choice>
    <xs:element name="ByNumber">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Person_Number">
            <xs:annotation>
              <xs:documentation>The SSIN or RRN to search
for</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
              <xs:restriction base="per:person_number">
                <xs:length value="11"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="source" minOccurs="0">
            <xs:annotation>
              <xs:documentation>Specify in which register to do
the research.
Possible value are BCSS and RRNP</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="BCSS"/>
                <xs:enumeration value="RRNP"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="ByDetails">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="first_name" minOccurs="0">
            <xs:annotation>
              <xs:documentation>First name of the person to
look for</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:whiteSpace value="collapse"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="middle_name" minOccurs="0">
            <xs:annotation>
              <xs:documentation>Second first_name of the person
to look for</xs:documentation>
            </xs:annotation>
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:whiteSpace value="collapse"/>
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="last_name">
            <xs:annotation>
              <xs:documentation>Last name of the person to look
for</xs:documentation>
            </xs:annotation>
            <xs:simpleType>

```

```

        <xs:restriction base="xs:string">
            <xs:whiteSpace value="collapse"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
<xs:element name="birth_date">
    <xs:annotation>
        <xs:documentation>Birthdate of the person, may be
incomplete, possible combination are yyyy-mm-dd, yyyy-mm,
yyyy</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:attribute name="birth_year" use="required">
            <xs:simpleType>
                <xs:restriction base="xs:integer">
                    <xs:minInclusive value="0"/>
                    <xs:maxInclusive value="9999"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="birth_month">
            <xs:simpleType>
                <xs:restriction base="xs:integer">
                    <xs:minInclusive value="1"/>
                    <xs:maxInclusive value="12"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="birth_day">
            <xs:simpleType>
                <xs:restriction base="xs:integer">
                    <xs:minInclusive value="1"/>
                    <xs:maxInclusive value="31"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
    </xs:complexType>
</xs:element>
<xs:element name="gender_code" type="GenderCodeType"
minOccurs="0">
    <xs:annotation>
        <xs:documentation>Gender code: 0=unspecified,
1=male, 2=female</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="birth_date_tolerance"
minOccurs="0">
    <xs:annotation>
        <xs:documentation>Amount that the birthdate of
the result may differ from the specified one. Must be specified unless the
BirthDate field contains a complete date (in which case 0 is assumed). Unit
type depends on the given format of Birthdate: yyyyymmdd: days; yyyyymm: months;
yyyy: years</xs:documentation>
    </xs:annotation>
    <xs:simpleType>
        <xs:restriction base="xs:integer">
            <xs:whiteSpace value="collapse"/>
            <xs:minInclusive value="0"/>
            <xs:maxInclusive value="30"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
<xs:element name="max_return_data" minOccurs="0">

```

```

        <xs:annotation>
          <xs:documentation>If this number is too high, an
error is returned.</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:integer">
            <xs:whiteSpace value="collapse"/>
            <xs:minInclusive value="0"/>
            <xs:maxInclusive value="100"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:choice>
<xs:element ref="eid:eid"/>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
  <xs:schema xmlns:iso="http://fsb.belgium.be/common/isocodes"
xmlns:per="http://fsb.belgium.be/prove/person"
xmlns="http://fsb.belgium.be/fphp/v2_1/fphp390"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://fsb.belgium.be/fphp/v2_1/fphp390"
elementFormDefault="qualified">
    <xs:import namespace="http://fsb.belgium.be/prove/person"/>
    <xs:import namespace="http://fsb.belgium.be/common/isocodes"/>
    <xs:annotation>
      <xs:documentation xml:lang="en">Schema version: 1.5
        Date: 2007-07-10
        Author: Ignaz Wanders
        Release notes: added two optional fields holding any information
coming from a backend
service in case of an error. Because these fields
are optional,
the schema version 1.5 is compatible with the
previous version 1.4.

        Schema version: 1.4
        Date: 2007-04-25
        Author: Ignaz Wanders
        Release notes: added three optional free-format address fields to
cope with
unstructured foreign addresses. Because these fields
are optional,
the schema version 1.4 is compatible with the
previous version 1.3.

        Schema version: 1.3
        Date: 2006-11-07
        Author: Quentin Dallons</xs:documentation>
    </xs:annotation>
    <xs:simpleType name="t_SSIN">
      <xs:annotation>
        <xs:documentation>social security identification
number</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">

```

```

        <xs:length value="11"/>
        <xs:whiteSpace value="collapse"/>
        <xs:pattern value="\p{N}{11}"/>
    </xs:restriction>
</xs:simpleType>
<xs:element name="fphp390">
    <xs:complexType>
        <xs:sequence>
            <xs:choice>
                <xs:element name="person" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="source" minOccurs="0">
                                <xs:simpleType>
                                    <xs:restriction base="xs:string">
                                        <xs:enumeration value="BCSS"/>
                                        <xs:enumeration value="RRNP"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="nrrn" type="per:person_number"/>
                            <xs:element name="gender_code" minOccurs="0">
                                <xs:simpleType>
                                    <xs:restriction base="iso:gender">
                                        <xs:enumeration value="0"/>
                                        <xs:enumeration value="1"/>
                                        <xs:enumeration value="2"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="last_name" type="xs:string"/>
                            <xs:element name="first_name" type="xs:string"
minOccurs="0"/>
                            <xs:element name="middle_name" type="xs:string"
minOccurs="0">
                                <xs:annotation>
                                    <xs:documentation>Second and following
firstname</xs:documentation>
                                </xs:annotation>
                            </xs:element>
                            <xs:element name="birth_day" minOccurs="0">
                                <xs:simpleType>
                                    <xs:restriction base="xs:positiveInteger">
                                        <xs:minInclusive value="1"/>
                                        <xs:maxInclusive value="31"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="birth_month" minOccurs="0">
                                <xs:simpleType>
                                    <xs:restriction base="xs:positiveInteger">
                                        <xs:minInclusive value="1"/>
                                        <xs:maxInclusive value="12"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="birth_year">
                                <xs:simpleType>
                                    <xs:restriction base="xs:positiveInteger">
                                        <xs:pattern value="\d\d\d\d"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                        </xs:sequence>
                    </xs:complexType>
                </xs:choice>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

```



```

        <xs:element name="ad_country_name" type="xs:string"
minOccurs="0" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="error">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="id" type="xs:string"/>
            <xs:element name="code" type="xs:string"/>
            <xs:element name="description" type="xs:string"/>
            <!-- added two optional fields to hold information about an
error
                                from the backend -->
            <xs:element name="backend_code" type="xs:string"
minOccurs="0" />
            <xs:element name="backend_description" type="xs:string"
minOccurs="0" />
            <xs:element name="suggested_action" type="xs:string"
minOccurs="0" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:choice>
<xs:element name="nb_calls">
    <xs:complexType>
        <xs:attribute name="BCSS_KSZ" type="xs:int"/>
        <xs:attribute name="RN_NR" type="xs:int"/>
    </xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
    <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:not="http://fsb.belgium.be/prove/notary"
xmlns:per="http://fsb.belgium.be/prove/person"
xmlns:ent="http://fsb.belgium.be/prove/enterprise"
xmlns:iso="http://fsb.belgium.be/common/isocodes"
targetNamespace="http://fsb.belgium.be/prove/notary"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <xs:annotation>
        <xs:documentation xml:lang="en">Schema version: 1.1
Date: 2005-07-26
Author: Ignaz Wanders</xs:documentation>
    </xs:annotation>
    <xs:import namespace="http://fsb.belgium.be/common/isocodes"/>
    <xs:import namespace="http://fsb.belgium.be/prove/person"/>
    <xs:import namespace="http://fsb.belgium.be/prove/enterprise"/>
    <xs:element name="notary">
        <xs:annotation>
            <xs:documentation xml:lang="en">The general identification of a
notary.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="office_id" type="xs:string">
                    <xs:annotation>
                        <xs:documentation xml:lang="en">The general ID of the Notary.
It is this ID that the FSB will use to identify a notary.</xs:documentation>
                    </xs:annotation>
                </xs:element>
                <xs:element name="lang" type="iso:language">

```

```

        <xs:annotation>
            <xs:documentation xml:lang="en">The language in which the
notary prefers to receive answers.
Note: KBO/BCE currently only recognizes "nl" and "fr" as valid
languages.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="nrn" type="per:person_number">
        <xs:annotation>
            <xs:documentation xml:lang="en">The national registry number of
the notary.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="num_kbo_not" type="ent:enterprise_number">
        <xs:annotation>
            <xs:documentation xml:lang="en">The KBO/BCE enterprise number
of the notary.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="num_kbo_fed" type="ent:enterprise_number">
        <xs:annotation>
            <xs:documentation xml:lang="en">The KBO/BCE enterprise number
of the federation of notaries.</xs:documentation>
        </xs:annotation>
    </xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
    <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:per="http://fsb.belgium.be/prove/person"
targetNamespace="http://fsb.belgium.be/prove/person"
elementFormDefault="qualified" attributeFormDefault="unqualified">
        <xs:annotation>
            <xs:documentation xml:lang="en">Schema version: 1.0 RC1
Date: 2005-04-08
Author: Ignaz Wanders</xs:documentation>
        </xs:annotation>
        <xs:simpleType name="person_number">
            <xs:annotation>
                <xs:documentation xml:lang="en">The type definition of the number of
a person.
                    It is based on a string to avoid problems with leading zeroes.

                    Validation rules:
                    - The length is 11 digits, of which the last two are control
digits.
                    - The first six digits are the birth date in the format yymmdd.
                    - Let num1 = number(0:9) and num2 = number(9:11)
                    - Then num2 = 97 - (num1 % 97) for birth dates up to 31/12/1999
                    and num2 = 97 - ( (num1 + 2*10^9) % 97 ) for birth dates
later than 31/12/1999

                    The modulus and birth date can not be captured in a regular
expression, but the
                    basic check on the digits and the length are used in a regular
expression to validate the enterprise number.

                    Note that birth dates are not always known, and exceptions to
the birth date rule
                    exist. Therefore, the birth date should not be considered in a
validation rule.</xs:documentation>
            </xs:annotation>

```

```

    <xs:restriction base="xs:string">
      <xs:pattern value="\d{11}"/>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
  <!-- edited with XML Spy v4.0.1 U (http://www.xmlspy.com) by Paul
Stijfhals (Recherche) -->
  <xs:schema xmlns:iso="http://fsb.belgium.be/common/isocodes"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://fsb.belgium.be/common/isocodes"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:annotation>
    <xs:documentation xml:lang="en">Schema version: 1.0 RC3
      Date: 2005-05-04
      Author: Ignaz Wanders</xs:documentation>
  </xs:annotation>
  <xs:simpleType name="language">
    <xs:annotation>
      <xs:documentation xml:lang="en">Language codes follow the ISO-639-1
two-letter code standards.
      For details, see http://www.loc.gov/standards/iso639-
2/iso639jac.html

```

Validation rules:
- two lower case letters

Regular expression: [a-z]{2}

Examples: nl, fr, de, en, es, ...

Notes.

- [a-zA-Z0-9]{1,8})*
- to denote regional
- three-letter code
- is not enforced
1. The xs:language type has a facet ([a-zA-Z]{1,8})(- which allows more than two characters. For example languages.
 2. For undetermined languages, ISO reserves the "und". For this reason, the two-letter restriction in the schema.</xs:documentation>

```

  </xs:annotation>
  <xs:restriction base="xs:language"/>
</xs:simpleType>
<xs:simpleType name="country">
  <xs:annotation>
    <xs:documentation xml:lang="en">Country codes must be in the ISO-3166
two-letter format.

```

For a complete list, see
<http://www.iso.org/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/index.html>

The FSB will translate the ISO code into specific KBO codes, if required.

The ISO standards permit certain two-letter codes to be customized by users. This allows for the following codes, which can be used withing the PROVE application:

	ISO CODE	KBO CODE	meaning
nationalities)	XA	900	stateless (when applied to
nationalities)	XB	901	not yet proven (when applied to

nationalities)	XC	992	moved to abroad (when applied
	XD	995	at sea (international waters)
	XE	999	undetermined

Validation rules:
- two upper case letters

Regular expression: [A-Z]{2}

Examples: BE, NL, FR, DE, GB, ES, ...

(Note that UK is not a valid country code!)</xs:documentation>

</xs:annotation>

<xs:restriction base="xs:string">

<xs:pattern value="[A-Z]{2}"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="currency">

<xs:annotation>

<xs:documentation xml:lang="en">Currency codes must follow the ISO-4217 code standards. These are three-letter codes

derived from the ISO-639-1 two-letter country codes. For

details, see

<http://www.bsi->

[global.com/British_Standards/currency/index.xalter](http://www.bsi-global.com/British_Standards/currency/index.xalter)

Validation rules:
- three upper case letters

Regular expression: [A-Z]{3}

Examples: EUR, GBP, USD, ...</xs:documentation>

</xs:annotation>

<xs:restriction base="xs:string">

<xs:pattern value="[A-Z]{3}"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="bic">

<xs:annotation>

<xs:documentation xml:lang="en">The type definition of a "Bank Identifier Code" (BIC), specified in ISO 9362.

For details, see

http://www.swift.com/biconline/index.cfm?fuseaction=display_aboutbic

Validation rules:
- The length is 8 or 11 characters.
- First 4 chars are alphabetic and denote the bank code
- 5th and 6th char are an ISO country code
- 7th and 8th char are alphanumeric and denote the region
within a country
- 9th - 11th char are the alphanumeric branch code
- A BIC code must be in upper case letters

Regular expression: [A-Z]{6}[A-Z0-9]{2}([A-Z0-9]{3}){0,1}

Examples: ABNAFRPP, GEBABEBB04A, ...</xs:documentation>

</xs:annotation>

<xs:restriction base="xs:string">

<xs:pattern value="[A-Z]{6}[A-Z0-9]{2}([A-Z0-9]{3}){0,1}"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="iban">

```

<xs:annotation>
  <xs:documentation xml:lang="en">The type definition of an
international bank account number (IBAN), specified in ISO 13616.

  Validation rules:
  - The length is up to 34 characters
  - The first two characters are the ISO two-letter country code
  - The 3rd and 4th character are numeric control digits
  - The 5th to the last char are alphanumeric
  - For the calculation of the control digits:
    o move the first four chars to the end of the number
    o convert each alphabetic char in the number to a digit
according to a conversion
    table: A=10, B=11, C=12, ..., Y=34, Z=35
    Note: each letter is converted to two digits, so the
number of chars increases
    o calculate the mod 97 of the full number: it must be
equal to one

  Regular expression: [A-Z]{2}\d{2}[A-Z0-9]{1,30}

  Examples: BE68539007547034, FR1420041010050500013M02606,
GB29NWBK60161331926819</xs:documentation>
</xs:annotation>
<xs:restriction base="xs:string">
  <xs:pattern value="[A-Z]{2}\d{2}[A-Z0-9]{1,30}"/>
</xs:restriction>
</xs:simpleType>
<xs:simpleType name="gender">
  <xs:annotation>
    <xs:documentation xml:lang="en">The type definition for a gender must
follow the ISO 5128 specification
    The following data items and codes are used
    Not known      0
    Male           1
    Female         2
    Not specified 9</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:integer">
    <xs:maxInclusive value="2"/>
    <xs:minInclusive value="0"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>
  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:eid="http://fsb.belgium.be/common/eid"
targetNamespace="http://fsb.belgium.be/common/eid"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:element name="eid">
    <xs:annotation>
      <xs:documentation>Comment describing your root
element</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="certificate" type="xs:string" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ent="http://fsb.belgium.be/prove/enterprise"
xmlns:iso="http://fsb.belgium.be/common/isocodes"

```

```

xmlns:adr="http://fsb.belgium.be/prove/address"
xmlns:per="http://fsb.belgium.be/prove/person"
targetNamespace="http://fsb.belgium.be/prove/enterprise"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:annotation>
    <xs:documentation xml:lang="en">Schema version: 1.2
      Date: 2005-07-26
      Author: Ignaz Wanders
        - removed all type definitions that are not used in this
project to minimize WSDL size

          Schema version: 1.1
          Date: 2005-07-26
          Author: Ignaz Wanders</xs:documentation>
    </xs:annotation>
    <xs:simpleType name="enterprise_number">
      <xs:annotation>
        <xs:documentation xml:lang="en">The type definition of the number of
an enterprise.

          It is based on a string to avoid problems with leading zeroes.

          Validation rules:
          - The length is 10 digits, of which the last two are control
digits.

          - The first digit is either a zero or a one.
          - Let num1 = number(0:8) and num2 = number(8:10)
          - Then num2 = 97 - (num1 % 97)

          The modulus can not be captured in a regular expression, but
the
          basic check on the digits and the length are used in a regular
expression to validate the enterprise
number.</xs:documentation>
      </xs:annotation>
      <xs:restriction base="xs:string">
        <xs:pattern value="[01]\d{9}"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:schema>

</types>
<message name="fphp100SoapIn">
  <part name="parameters" element="s0:fphp100"/>
</message>
<message name="fphp100SoapOut">
  <part name="parameters" element="s0:fphp100Response"/>
</message>
<message name="pingSoapIn">
  <part name="parameters" element="s0:ping"/>
</message>
<message name="pingSoapOut">
  <part name="parameters" element="s0:pingResponse"/>
</message>
<message name="testSOAPFaultSoapIn">
  <part name="parameters" element="s0:testSOAPFault"/>
</message>
<message name="testSOAPFaultSoapOut">
  <part name="parameters" element="s0:testSOAPFaultResponse"/>
</message>
<message name="pingHttpGetIn"/>
<message name="pingHttpGetOut">
  <part name="Body" element="s0:string"/>
</message>

```

```

<message name="testSOAPFaultHttpGetIn"/>
<message name="testSOAPFaultHttpGetOut"/>
<message name="pingHttpPostIn"/>
<message name="pingHttpPostOut">
  <part name="Body" element="s0:string"/>
</message>
<message name="testSOAPFaultHttpPostIn"/>
<message name="testSOAPFaultHttpPostOut"/>
<portType name="findPersonSoap">
  <operation name="fphp100">
    <input message="s0:fphp100SoapIn"/>
    <output message="s0:fphp100SoapOut"/>
  </operation>
  <operation name="ping">
    <input message="s0:pingSoapIn"/>
    <output message="s0:pingSoapOut"/>
  </operation>
  <operation name="testSOAPFault">
    <input message="s0:testSOAPFaultSoapIn"/>
    <output message="s0:testSOAPFaultSoapOut"/>
  </operation>
</portType>
<portType name="findPersonHttpGet">
  <operation name="ping">
    <input message="s0:pingHttpGetIn"/>
    <output message="s0:pingHttpGetOut"/>
  </operation>
  <operation name="testSOAPFault">
    <input message="s0:testSOAPFaultHttpGetIn"/>
    <output message="s0:testSOAPFaultHttpGetOut"/>
  </operation>
</portType>
<portType name="findPersonHttpPost">
  <operation name="ping">
    <input message="s0:pingHttpPostIn"/>
    <output message="s0:pingHttpPostOut"/>
  </operation>
  <operation name="testSOAPFault">
    <input message="s0:testSOAPFaultHttpPostIn"/>
    <output message="s0:testSOAPFaultHttpPostOut"/>
  </operation>
</portType>
<binding name="findPersonSoap" type="s0:findPersonSoap">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document"/>
  <operation name="fphp100">
    <soap:operation soapAction="http://fsb.belgium.be/prove/fphp100"
style="document"/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
  <operation name="ping">
    <soap:operation soapAction="http://fsb.belgium.be/prove/ping"
style="document"/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>

```

```

        </output>
    </operation>
    <operation name="testSOAPFault">
        <soap:operation soapAction="http://fsb.belgium.be/prove/testSOAPFault"
style="document" />
        <input>
            <soap:body use="literal" />
        </input>
        <output>
            <soap:body use="literal" />
        </output>
    </operation>
</binding>
<binding name="findPersonHttpGet" type="s0:findPersonHttpGet">
    <http:binding verb="GET" />
    <operation name="ping">
        <http:operation location="/ping" />
        <input>
            <http:urlEncoded />
        </input>
        <output>
            <mime:mimeType part="Body" />
        </output>
    </operation>
    <operation name="testSOAPFault">
        <http:operation location="/testSOAPFault" />
        <input>
            <http:urlEncoded />
        </input>
        <output />
    </operation>
</binding>
<binding name="findPersonHttpPost" type="s0:findPersonHttpPost">
    <http:binding verb="POST" />
    <operation name="ping">
        <http:operation location="/ping" />
        <input>
            <mime:contentType type="application/x-www-form-urlencoded" />
        </input>
        <output>
            <mime:mimeType part="Body" />
        </output>
    </operation>
    <operation name="testSOAPFault">
        <http:operation location="/testSOAPFault" />
        <input>
            <mime:contentType type="application/x-www-form-urlencoded" />
        </input>
        <output />
    </operation>
</binding>
<service name="findPerson">
    <port name="findPersonSoap" binding="s0:findPersonSoap">
        <soap:address
location="http://fsb.belgium.be:80/fphp/2.1/ws/findPerson.jws" />
    </port>
    <port name="findPersonHttpGet" binding="s0:findPersonHttpGet">
        <http:address
location="http://fsb.belgium.be:80/fphp/2.1/ws/findPerson.jws" />
    </port>
    <port name="findPersonHttpPost" binding="s0:findPersonHttpPost">
        <http:address
location="http://fsb.belgium.be:80/fphp/2.1/ws/findPerson.jws" />

```

```
</port>  
</service>  
</definitions>
```