EN

Installation and Configuration

# AEOS SOAP WebService

Version 33 | 10-07-2020

nedap | security management

www.nedapsecurity.com

| Date | Version | Changes |
|---|---|---|
| 10-07-2020 | 33 | Added Event types.<br>Added notes that SOAP does not support adding or changing OSS-SO or OffLine Entrances or Entrance Groups.<br>Added *removeCarrierAuthorizations* to the Authorization functions list.<br>Removed Updates 1 to 28 from this table. |
| 25-06-2020 | 32 | Minor corrections |
| 27-09-2019 | 31 | Corrected Rename the WebService URL<br>Removed '-1' as option for an empty object ID from the *Change* paragraph in the General entity handling section because this does not work for all functions, while '0' (the other option) does work for all functions. |
| 29-03-2019 | 30 | Added findAPBZone, findEntranceZone and findCarriersPresence.<br>Added Date and time fields.<br>Changed all **AEserver\jboss** folders to **AEserver**. |
| 25-02-2019 | 29 | Added code examples and the AEOS terminology overview.<br>Minor corrections in 'modify system users in AEOS to handle SOAP calls'<br>WSDL address is now only https instead of http, and added suggestion to use a dedicated WSDL viewer such as SoapUI.<br>Added remark to 'General entity handling' section. |

# Contents

# 1. Set up SOAP in AEOS

With the AEOS SOAP WebService, third parties can receive and change data in AEOS with SOAP calls.

This chapter describes how to get SOAP and how to connect it to AEOS.

## 1.1 Enable SOAP in AEOS

### AEOS Blue

1. During the AEOS software installation (setup), select **SOAP WebService** in the **Select the desired options for the AEOS application** window. If the AEOS is already installed, just run the setup again and don't change any settings except the selection of the SOAP WebService.
2. In AEOS, enable SOAP in the system properties.
   a. go to go to **Administration > Maintenance > Settings > System properties**.
   b. Select the **SOAP WebService** checkbox (44.15).
   c. Click **OK**.
   d. Restart the AEOS application server service (see 6.2).

### AEOS Classic

1. Purchase license option 8019223.
2. During the AEOS software installation (setup), activate option **8019223 SOAP WebService**.

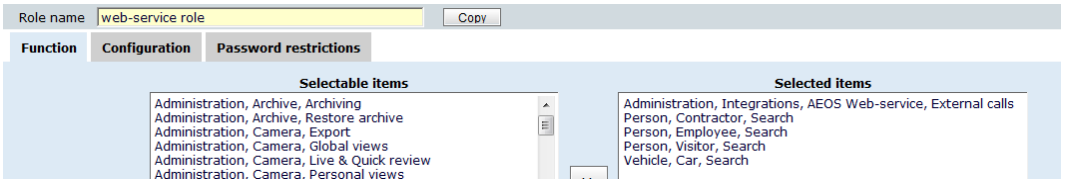
## 1.2 Create or modify system users in AEOS to handle SOAP calls

The program that sends SOAP calls to AEOS, needs to 'log in' to AEOS as a system user in order to have the rights to access or change information there. For this reason, you need to create one or more system users in AEOS and give these users a user role with the correct authorities to 'access the correct menu options' from within AEOS.

All user actions (issuing badges, deleting persons etc.) will be logged on the system user account that is used to connect. For that reason, it is preferred that all system users that log in though SOAP have their own account, otherwise you will not be able to determine who made a specific change in the system.

**For more information on system users, user roles and filters in AEOS, see the *Create and manage system users* chapter in the *AEOS User manual* version 3.4.0.x or higher, available on the Nedap web portal via https://portal.nedapsecurity.com/document/aeos_user_manual_english.**

1. Log in to AEOS.
2. Create a user role (or edit an existing one) for the SOAP client, and select at least these functions:
   - **Administration, Integrations, AEOS WebService, External calls**.
   - **Person, Contractor, Search**.
   - **Person, Employee, Search**.
   - **Person, Visitor, Search**.
   - **Vehicle, Car, Search**.




The first function is necessary to make WebService calls. The other functions allow find/add/update/delete actions on contractors, employees, visitors and/or cars.

3. If necessary, define an *Entrance filter* to restrict the entrances that can be seen, changed, deleted, or used for carrier authorization by the SOAP client.
4. Create or modify system users for use with the SOAP client.
   - If you will use only one system user for the SOAP client, create this user and assign them the correct user role and any filters you may have defined.
   - If people will log in with their own system user account through SOAP, add the correct functions and filters to their existing user role.

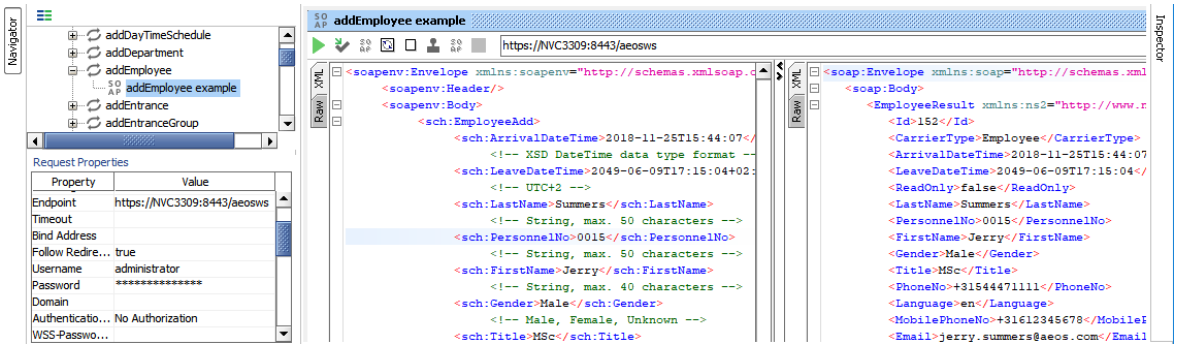
## 1.3 Connect to the AEOS SOAP web interface

AEOS uses basic HTTP authentication for SOAP system user logins. The username and password is part of the http request.

- SOAP address location:
  `https://<servername>:8443/aeosws`
- WSDL function definitions:
  `https://<servername>:8443/aeosws?wsdl`

  The WSDL file **AeosWebService.wsdl** and schema definition file **AeosWebService.xsd** are also stored in the **\AEserver\standalone\deployments.aeos\aeos-server.ear\aeosws.jar**[1] file on the AEOS server.

With a tool such as *soapUI* you can test the AEOS WebService using http authentication, and see all the functions that are defined in the WSDL file.




[1] *Up to AEOS 3.4, the AEserver\standalone folder was located in AEserver\jboss\standalone.*

## 1.4 Rename the WebService URL

If the AEOS server is placed behind a firewall and IP-forwarding is used, the server could have a different external and internal name. For example, the external name could be *AEOSProduction*, while internally the server is named *AEOS1234*.

When you open the WSDL function definition page on the *AEOSProduction* server, you will receive the WSDL definition with this soap address:

```
<soap:address location= "AEOSProduction:8443/aeosws".
```

If the return soap address must be the actual server *AEOS1234* in the http request, do as follows.

1. Open the **AEserver\standalone\configuration\standalone.xml**[2] file with a text editor.
2. In this file, find this line:

   ```
   <wsdl-host>${jboss.bind.address:127.0.0.1}</wsdl-host>
   ```

3. Replace the contents of the tag with the hostname that you want to use for your server (in this example *AEOS1234*):

   ```
   <wsdl-host>AEOS1234</wsdl-host>
   ```

## 1.5 Enable TLS encryption

This is described in the *AEOS Advanced Installation & Configuration* manual.

[2] *Up to AEOS 3.4, the AEserver\standalone folder was located in AEserver\jboss\standalone.*

# 2. Functions

## 2.1 WSDL function definitions

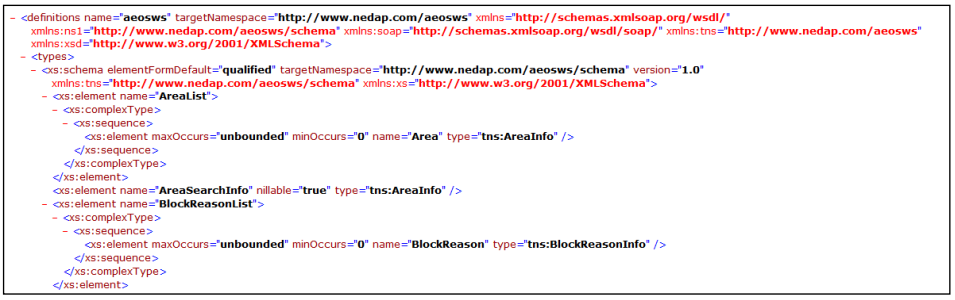
WSDL (Web Services Description Language) is the XML language that the AEOS SOAP WebService uses to connect.

The AEOS WebService is completely defined in this WSDL, including:

- all supported functions
- fields in the functions
- field properties (type / size / cardinality / enumeration / ...)

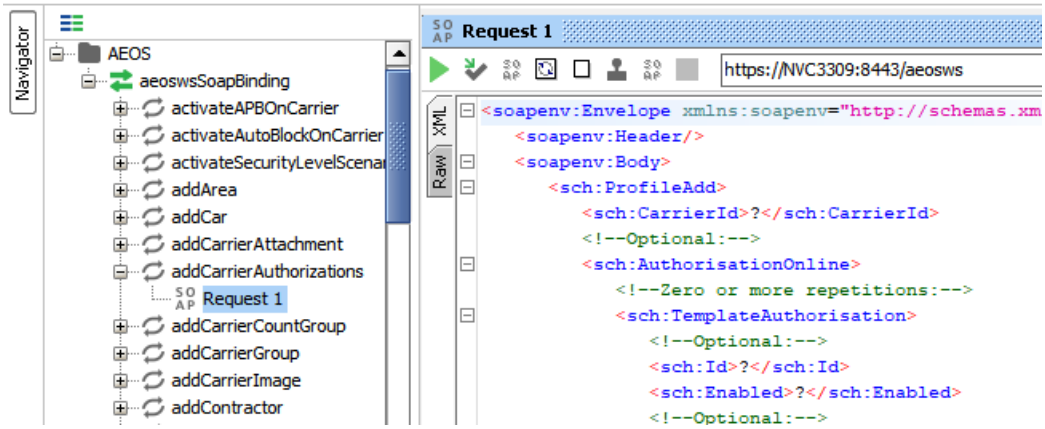
For more information about WSDL, see https://en.wikipedia.org/wiki/Web_Services_Description_Language and https://www.w3schools.com/xml/xml_wsdl.asp.

1. To see the WSDL function definitions for AEOS, open this web address on the AEOS server: `https://<servername>:8443/aeosws?wsdl`




The WSDL file **AeosWebService.wsdl** and schema definition file **AeosWebService.xsd** are also stored in the **\AEserver\standalone\deployments.aeos\aeos-server.ear\aeosws.jar**[3] file on the AEOS server.

2. If you open the web address with a tool such as *soapUI*, you can see the defined functions in a more readable way.




[3] *Up to AEOS 3.4, the AEserver\standalone folder was located in AEserver\jboss\standalone.*

## 2.2 Available SOAP functions

To find out what the terms *carrier*, *access point* and *entrance* etc. mean, see AEOS terminology (6.1).

For an overview of what kind of operations can and *cannot* be done with SOAP, see also Basic AEOS operations in SOAP (chapter 3) and Configure an AEOS system with SOAP (chapter 4).

### 2.2.1 Carrier functions

- findPerson (3.1.3), findPersonId, findCarrierByToken
- findEmployee, addEmployee (3.1.1), changeEmployee, removeEmployee (3.1.5)
- findContractor, addContractor, changeContractor, removeContractor
- findVisitor, addVisitor, changeVisitor, removeVisitor
- findCar, addCar (3.1.2), changeCar, removeCar
- findCarrierImage, addCarrierImage, removeCarrierImage
- findCarrierAttachment, addCarrierAttachment, changeCarrierAttachment, removeCarrierAttachment
- changeCarrierAttribute
- blockCarrier (3.1.6), unblockCarrier (3.1.8), findBlockReason
- assignContactPersons, withdrawContactPersons
- findCarrierPresence, findCarriersPresence
- findCarrierStates (3.1.9)

#### Visit management / vendor management (optional)

- findPermit, addPermit, changePermit, withdrawPermit, removePermit
- findVendor, addVendor, changeVendor, removeVendor
- findVisit, addVisit, changeVisit, removeVisit

### 2.2.2 Token/identifier functions

Tokens are called *identifiers* or *badges* in AEOS

- findToken, findCarrierToken (3.2.2)
- findIdentifierType
- addToken
- assignToken (3.2.1), withdrawToken (3.2.4)
- replaceToken (3.2.3)
- blockToken (3.2.5), unblockToken (3.2.6.2), findBlockReason
- withdrawAndBlockToken, releaseBlockedToken (3.2.6.3)

### 2.2.3 Access Point / Entrance functions

- findEntrance (4.3.2), addEntrance* (4.3.1), changeEntrance* (4.3.3), removeEntrance*
- findUnconfirmedAccessPoint (4.4.1), confirmAccessPoints (4.4.2)
- findAccessPoint (4.4.3), changeAccessPoint (4.4.4)
- findEntranceGroup (4.5.2), addEntranceGroup* (4.5.1), changeEntranceGroup* (4.5.3), removeEntranceGroup*

*SOAP does not support adding, changing or removing OSS-SO (Soaa) or OffLine Entrances or Entrance Groups.

#### Entrance / entrance group labels

- findCountry (4.1.2), addCountry (4.1.1), changeCountry, removeCountry
- findRegion, addRegion (4.2.1), changeRegion, removeRegion
- findArea, addArea (4.2.1), changeArea, removeArea
- findOrganization, addOrganization (4.2.1), changeOrganization, removeOrganization

### 2.2.4 Authorization functions

- findDayTimeSchedule (4.6.2), addDayTimeSchedule (4.6.1), changeDayTimeSchedule (4.6.3), removeDayTimeSchedule
- findHoliday, addHoliday (4.7), changeHoliday, removeHoliday
- findTemplate (4.9.2), addTemplate (4.9.1), changeTemplate (4.9.3), removeTemplate
- findCarrierProfile, changeCarrierProfile (3.1.4)
- addCarrierAuthorizations, removeCarrierAuthorizations
- findCarrierVerification, changeCarrierVerification
- activateAutoBlockOnCarrier (3.1.7), deactivateAutoBlockOnCarrier
- activateAPBOnCarrier, deactivateAPBOnCarrier, findAPBZone (Anti Pass Back, optional)
- findDepartment, addDepartment, changeDepartment, removeDepartment
- findUnit, addUnit, changeUnit, removeUnit (optional)

### 2.2.5 Free fields

- findFreeFieldCategory
- findFreeFieldDefinition

### 2.2.6 Counting functions (optional)

- findCountZoneManager
- findCountZone, addCountZone, changeCountZone, removeCountZone
- findCarrierCountGroup, addCarrierCountGroup, removeCarrierCountGroup
- findCountGroup, addCountGroup, changeCountGroup, removeCountGroup
- findCountZoneCountGroupAmount, setCountZoneCountGroupAmount
- findCountZoneCountGroupConfiguration, configureCountZoneCountGroup
- findEntranceZone (for *Maximum presence time* and *Maximum number of movements*, optional)

### 2.2.7 Security Level functions (optional)

- findSecurityLevelTemplate, addSecurityLevelTemplate, changeSecurityLevelTemplate, removeSecurityLevelTemplate
- findSecurityLevelType, addSecurityLevelType, changeSecurityLevelType, removeSecurityLevelType
- findSecurityLevelScenario, addSecurityLevelScenario, changeSecurityLevelScenario, removeSecurityLevelScenario
- activateSecurityLevelScenario
- findCarrierGroup, addCarrierGroup, changeCarrierGroup, removeCarrierGroup
- findCarrierGroupIdOfCarrier
- setCarrierGroupOnCarriers, resetCarrierGroupOnCarriers

### 2.2.8 Event / log functions

- findEvent
- findEventType
- findLog
- findLogType
- setCarrierSpecial, resetCarrierSpecial
- setCarrierInvisible, resetCarrierInvisible

See also Event types (chapter 5)

### 2.2.9 Locker functions (optional)

- findFreeLoXSLocker
- findLoXSTerminal

### 2.2.10 System functions

- getVersion

## 2.3 Enumerations

Some fields defined in the WSDL are enumerations. The used enumerations and their values can be found in the AeosWebService.xsd file. stored in the **\AEserver\standalone\deployments.aeos\aeos-server.ear\aeosws.jar**[4] file on the AEOS server.

An example of the used enumerations are:

| Field | Values |
|---|---|
| ae_int_DayOfWeek | 0(=Mo.), 1(=Tu), 2, 3, 4, 5, 6(=Su.) |
| AuthSubject | Template, EntranceGroup, Entrance |
| CarrierState | ActivateAutoblock, Block, ActivateApb, ActivateVerification, Invisible, Special. |
| GenderInfo | Unknown, Male, Female |
| <Identifier><Status> | 0 = Free, 1 = In use, 2 = Permanently blocked, 3 = Temporarily blocked, 4 = This badge temporarily replaces another badge. |
| PictureType | Photo, Thumbnail |
| ScheduleType | Week, Free |
| TypeOfCarrier | Employee, Visitor, Carrier, Car, Contractor, Person, Vehicle |
| <UnitOfAuthType> | OnLine, OffLine, SecurityLevel (Entrance Group), Soaa (OSS-SO), Loxs (Locker) |
| VerificationType | Pincode, Sagem Fingerprint<br>(Not yet implemented is Hitachi Fingervein, Weight) |

## 2.4 General entity handling

All objects in AEOS have a unique ID within their type. AEOS generates this object ID for all objects, the moment the object is created. This object ID is also present in the WebService.

### 2.4.1 Add

In the 'add' functions, the object is not used. AEOS generates the ID, not the caller. All 'add' functions return the created object with the generated ID.

### 2.4.2 Change

In the 'change' functions, the object ID is the key. When the object is found, all given fields will be updated in AEOS. All 'change' functions return the changed object.

**The personnel number is not the key (like it was in the import table).**

**To empty or deactivate a field that must contain an object ID, use the value '0' instead of leaving it empty. For example, when you want to remove an automatic unlock schedule from an entrance, and similar changes.**

[4] *Up to AEOS 3.4, the AEserver\standalone folder was located in AEserver\jboss\standalone.*

### 2.4.3 Remove

The 'remove' functions only require the object ID.

### 2.4.4 Find

In the 'find' functions, you can define a filter. All find functions support the search on ID and Name, but might not support other fields. For example, in the find method on carrier types, the fields: Gender, Title, PhoneNo, Guard, Language, MobielPhone, Email have not been implemented. Search on "String" fields will be executed as a "like 'input%'"on the Database, as it does in the AEOS front end.

## 2.5 How WebService functions work

Each WebService function is based on calls, parameters that specify the extent of the call, and responses.

For example, the client makes a **findVisitor** call to the WebService. If the client wants to get information about a visitor with the last name *James*, it specifies this parameter as follows.

```
<sch:LastName>James</sch:LastName>
```

The WebServer searches in the AEOS database for a visitor with the last name *James*. When that name is found, information about that visitor will be sent back to the client.

### 2.5.1 Parameters

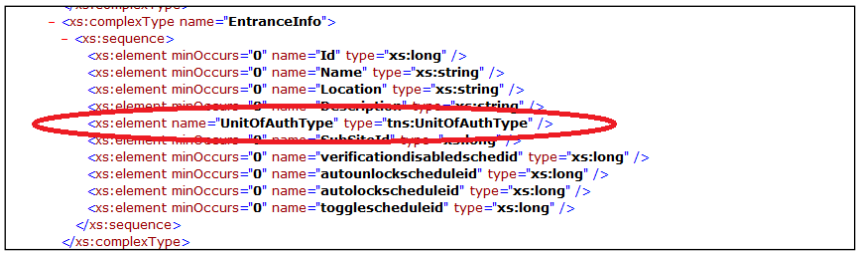
The WebServer only looks for the search parameters that are specified, and ignores all parameters that are disabled or not present.

In the previous example, the WebService looks for visitors with last name *James*, and returns a complete list of all visitors with this last name, independent of their first name, because the first name was not specified.

### 2.5.2 Mandatory fields

Some calls have a mandatory field. This is defined in the **AeosWebService.xsd** file.

In the example below, all fields with **minOccurs="0"** are optional.
The field **UnitOfAuthType** does not have this parameter, so this field *must* be added.



### 2.5.3 Date and time fields

The AEOS date and time fields in SOAP are in the *XSD Date* or *XSD DateTime* data type format, for example **2018-11-25T13:30:00**

- To create an empty date or delete an already existing date, enter **2149-05-29T00:00:00**

For more information about the XSD DateTime format, see https://www.w3schools.com/xml/schema_dtypes_date.asp.

### 2.5.4 Authorisations that were assigned by the rule engine

If the *rule engine* in AEOS (see the *AEOS user manual*) has automatically issued an authorisation to a carrier, you can see this in the **<RuleId>** field.

```
<CarrierId>1491090</CarrierId>
<AuthorisationOnline>
    <TemplateAuthorisation>
        <Id>1862066</Id>
        <Enabled>true</Enabled>
        <RuleId>5</RuleId>
        <TemplateId>43040</TemplateId>
        <DateFrom>2018-11-25T00:00:00</DateFrom>
    </TemplateAuthorisation>
</AuthorisationOnline>
```

*The rule engine has automatically assigned this authorisation*

You can see if an authorisation was issued by a rule, but you cannot add or change a rule authorisation with the WebService, because the rule engine itself determines these authorisations. The field **<RuleId>** will be ignored in the **addCarrierAuthorization** and **changeCarrierProfile** functions.

### 2.5.5 Return only a specified number of records

If you expect that a call returns a large number of items, you can specify the maximum amount of records to return in many *find* functions.

For example, the request below returns 10 records starting by the first record.

```
<sch:SearchRange>
    <sch:startRecordNo>0</sch:startRecordNo>
    <sch:nrOfRecords>10</sch:nrOfRecords>
</sch:SearchRange>
```

*Example code to limit the number of records that a find call can return*

When the amount of found records is less than 10, for example, 7, then 7 records will be returned instead of 10.

If **startRecordNo** is set to **3**, then the first 3 records will be skipped.

If **startRecordNo** has a higher value than **nrOfRecords**, no record will be returned.

# 3. Basic AEOS operations in SOAP

This section gives an overview of day-to-day AEOS procedures that you can do with SOAP.

For information on how to set up an AEOS system with SOAP, see Configure an AEOS system with SOAP (chapter 4) instead.

**To find out what the terms *carrier, access point, entrance* etc. mean, see AEOS terminology (6.1).**

### People or cars (carriers)

- Add a visitor, employee, or contractor to the system (3.1.1).
- Add a car to the system (3.1.2).
- Find someone (3.1.3).
- Give someone authorisation to access an area (3.1.4).
- Delete someone (3.1.5).
- Block someone (3.1.6).
- Block someone automatically after a period of inactivity (3.1.7).
- Unblock someone (3.1.8).
- Find out someone's status (blocked, APB, special, etc.) (3.1.9).

### Badges (identifiers, tokens)

- Give someone a badge - assign a token (3.2.1).
- Find out which badges (identifiers, tokens) someone has (3.2.2).
- Replace a forgotten, lost, stolen, or defective badge (3.2.3).
- Take back a badge - withdraw a token (3.2.4).
- Block a badge temporarily (3.2.5).
- Unblock a badge (3.2.6).

**Most code examples do not include the SOAP envelope and header code, which is the same for every call. Make sure to add this code to your own call if you copy the code example directly into you own application. See the add a person example (3.1.1) for a code example where the envelope and header code is included.**

### Basic operations in AEOS that are not possible with SOAP

- Open an entrance manually.
- Activate emergency entrances.

## 3.1 People or cars (carriers)

### 3.1.1 Add a visitor, employee, or contractor to the system

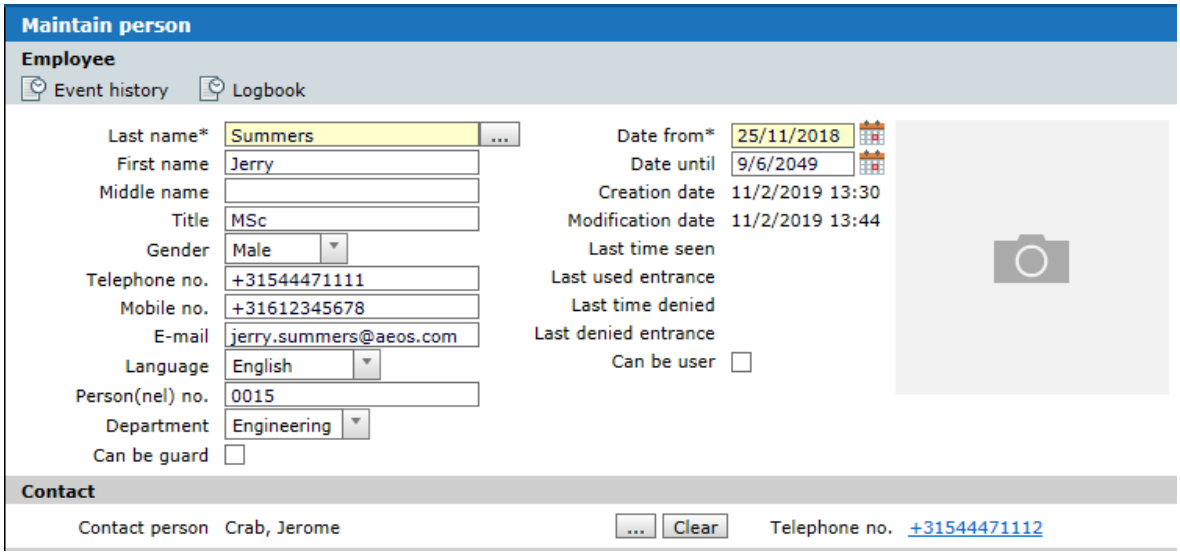
- Use the function **addEmployee** / **addVisitor** / **addContractor**.
- Only the **LastName** field is mandatory.
  Depending on the system configuration, more fields might be mandatory.
- If the **ID** field is added with a specified value, AEOS ignores it, because AEOS generates the ID, not the caller. All 'add' functions return the created object with the generated ID.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.nedap.com/aeosws/schema">
    <soapenv:Header/>
    <soapenv:Body>
        <sch:EmployeeAdd>
            <sch:ArrivalDateTime>2018-11-25T15:44:07</sch:ArrivalDateTime>
                <!-- XSD DateTime data type format -->
            <sch:LeaveDateTime>2049-06-09T17:15:04+02:00</sch:LeaveDateTime>
                <!-- UTC+2 -->
            <sch:LastName>Summers</sch:LastName>
                <!-- String, max. 50 characters -->
            <sch:PersonnelNo>0015</sch:PersonnelNo>
                <!-- String, max. 50 characters -->
            <sch:FirstName>Jerry</sch:FirstName>
                <!-- String, max. 40 characters -->
            <sch:Gender>Male</sch:Gender>
                <!-- Male, Female, Unknown -->
            <sch:Title>MSc</sch:Title>
                <!-- String, max. 25 characters -->
            <sch:PhoneNo>+31544471111</sch:PhoneNo>
                <!-- String, max. 25 characters -->
                <!-- Do not use spaces or hyphens if dialers (such as SMS servers)
                    need to process this phone number. -->
            <sch:Language>en</sch:Language>
                <!-- ar, de, dk, en, es, fa, fr, it, iw, nl, no, pl, pt, ru, sv,zh_CN -->
                <!-- String, max. 12 characters (but most are unused) -->
            <sch:MobilePhoneNo>+31612345678</sch:MobilePhoneNo>
                <!-- String, max. 25 characters -->
            <sch:Email>jerry.summers@aeos.com</sch:Email>
                <!-- String, max. 128 characters -->
            <sch:ContactPersonId>60</sch:ContactPersonId>
                <!-- Long. You can find this ID with findPerson -->
            <sch:DepartmentId>2</sch:DepartmentId>
                <!-- Long. You can find this ID with findDepartment -->
        </sch:EmployeeAdd>
    </soapenv:Body>
</soapenv:Envelope>
```

*Example code for **addEmployee**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <EmployeeResult xmlns:ns2=http://www.nedap.com/aeosws
xmlns="http://www.nedap.com/aeosws/schema">
            <Id>152</Id>
            <CarrierType>Employee</CarrierType>
            <ArrivalDateTime>2018-11-25T15:44:07</ArrivalDateTime>
            <LeaveDateTime>2049-06-09T17:15:04</LeaveDateTime>
            <ReadOnly>false</ReadOnly>
            <LastName>Summers</LastName>
            <PersonnelNo>0015</PersonnelNo>
            <FirstName>Jerry</FirstName>
            <Gender>Male</Gender>
            <Title>MSc</Title>
            <PhoneNo>+31544471111</PhoneNo>
            <Language>en</Language>
            <MobilePhoneNo>+31612345678</MobilePhoneNo>
            <Email>jerry.summers@aeos.com</Email>
            <ContactPersonId>60</ContactPersonId>
            <DepartmentId>2</DepartmentId>
        </EmployeeResult>
    </soap:Body>
</soap:Envelope>
```

*Response for the previous **addEmployee** code.*




*Result in AEOS.*

### 3.1.2 Add a car to the system

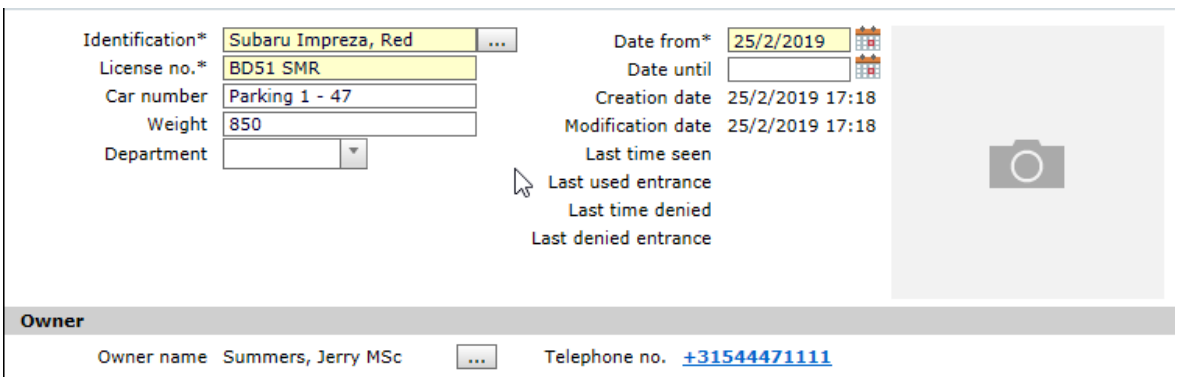
- Use the function **addCar**.
- The **Identification** and **LicenseNumber** fields are mandatory.
  Depending on the system configuration, more fields might be mandatory.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.nedap.com/aeosws/schema">
    <soapenv:Header/>
    <soapenv:Body>
        <sch:CarAdd>
            <sch:Identification>Subaru Impreza, Red</sch:Identification>
                <!-- String, max. 50 characters -->
            <sch:Weight>850</sch:Weight>
                <!-- Int -->
            <sch:LicenceNumber>BD51 SMR</sch:LicenceNumber>
                <!-- String, max. 50 characters -->
            <sch:CarNumber>Parking 1 - 47</sch:CarNumber>
                <!-- String, max. 25 characters -->
            <sch:OwnerId>152</sch:OwnerId>
                <!-- Long. You can find this ID with findPerson -->
        </sch:CarAdd>
    </soapenv:Body>
</soapenv:Envelope>
```

*Example code for **addCar**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <CarResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
            <Id>156</Id>
            <CarrierType>Car</CarrierType>
            <ArrivalDateTime>2019-02-25T17:18:08</ArrivalDateTime>
            <ReadOnly>false</ReadOnly>
            <Identification>Subaru Impreza, Red</Identification>
            <Weight>850</Weight>
            <LicenceNumber>BD51 SMR</LicenceNumber>
            <CarNumber>Parking 1 - 47</CarNumber>
            <OwnerId>152</OwnerId>
        </CarResult>
    </soap:Body>
</soap:Envelope>
```

*Response for the previous code.*




*Result in AEOS.*

### 3.1.3 Find someone

You can use this to find someone's person ID, which you need to edit their authorizations etc.

- Use the function **findPerson** / **findEmployee** / **findVisitor** / **findContractor** / **findCar**.
- There are no mandatory fields. However, if you do not specify any search criteria, the search will find everyone, which can take very long.
- If you expect that this call returns a large number of items, you can specify the maximum amount of records to return (2.5.5).

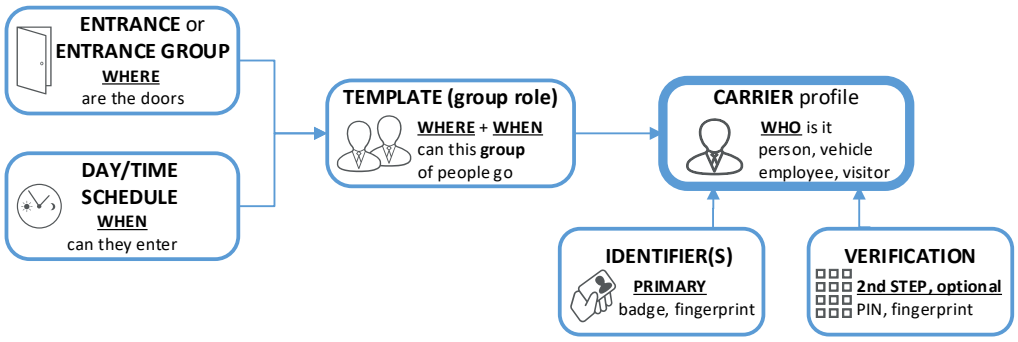
```
<soapenv:Body>
    <sch:PersonSearchInfo>
        <sch:PersonInfo>
            <sch:LastName>Summers/sch:LastName> <!-- leave this line out to find everyone -->
        </sch:PersonInfo>
    </sch:PersonSearchInfo>
</soapenv:Body>
```

*Example code for **findPerson**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <PersonList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
            <Person xsi:type="EmployeeInfo" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
                <Id>152</Id>
                <CarrierType>Employee</CarrierType>
                <ArrivalDateTime>2018-11-25T00:00:00</ArrivalDateTime>
                <LeaveDateTime>2049-06-09T00:00:00</LeaveDateTime>
                <ReadOnly>false</ReadOnly>
                <LastName>Summers</LastName>
                <PersonnelNo>0015</PersonnelNo>
                <FirstName>Jerry</FirstName>
                <Gender>Male</Gender>
                <Title>MSc</Title>
                <PhoneNo>+31544471111</PhoneNo>
                <Language>en</Language>
                <MobilePhoneNo>+31612345678</MobilePhoneNo>
                <Email>jerry.summers@aeos.com</Email>
                <ContactPersonId>60</ContactPersonId>
                <DepartmentId>2</DepartmentId>
            </Person>
        </PersonList>
    </soap:Body>
</soap:Envelope>
```

*Response for the previous code.*

### 3.1.4 Give someone authorisation to access an area




*Authorisation model in AEOS*

The collection of carrier authorizations is called a carrier profile.

A carrier profile can have these authorization types (UnitOfAuthType) :

- Online doors.
- Soaa (OSS-SO) doors.
- Offline doors.
- LoXS lockers.

For online doors, you normally use templates for authorization. OSS-SO doors do not support templates, so for these doors you need to assign entrances or entrance groups to a carrier directly, combined with a day/time schedule.

Each authorization (template, entrance, entrance group etc.) that is added to a carrier profile has its own ID inside that carrier profile.

See the AEOS terminology overview (6.1) for an explanation of Online, Soaa (OSS-SO) and Offline doors, security level management, LoXS locker management, entrances, templates, etc.

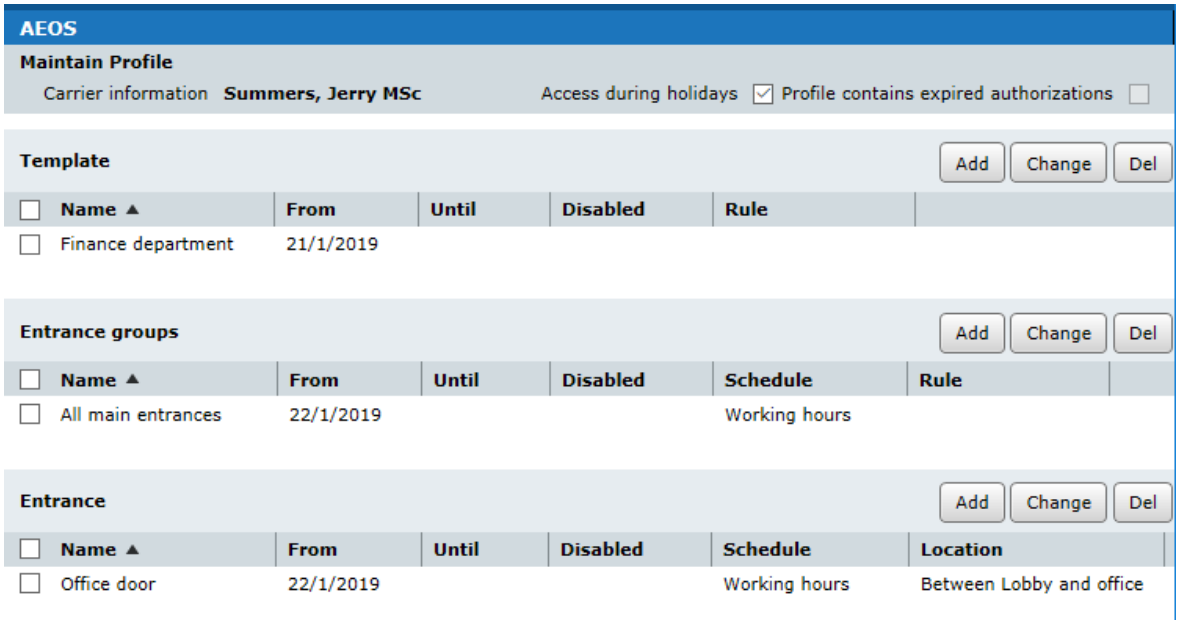
1. Find the ID of the carrier (3.1.3) that you want to give access.
2. Find the IDs of the templates (4.9.2), entrances (4.3.2), entrance groups (4.5.2) and day/time schedules (4.6.2) that you want to assign to this carrier..
3. To find out which authorisations someone already has, use the **findCarrierProfile** function.
4. To add authorisations to a profile without overwriting existing authorisations, use the **addCarrierAuthorizations** function.
5. To delete all existing authorisations of a carrier profile and replace them with new ones (or to give someone new authorizations for the first time), use the **changeCarrierProfile** function.
6. To change only part of a carrier profile, get all existing authorizations with **findCarrierProfile**, and then write everything back (including any changes) with **changeCarrierProfile**.

```
<soapenv:Body>
    <sch:ProfileChange>
        <sch:CarrierId>152</sch:CarrierId>
            <!-- Long. You can find this ID with findPerson -->
        <sch:AuthorisationOnline>
            <sch:TemplateAuthorisation>
                <sch:Enabled>true</sch:Enabled>
                <sch:TemplateId>51</sch:TemplateId>
                    <!-- Long. You can find this ID with findTemplate -->
                <sch:DateFrom>2019-01-21T00:00:00</sch:DateFrom>
                <sch:DateUntil>2149-05-29T00:00:00</sch:DateUntil>
                    <!-- XSD DateTime data type format -->
            </sch:TemplateAuthorisation>
            <sch:EntranceGroupAuthorisation>
                <sch:Enabled>true</sch:Enabled>
                <sch:EntranceGroupId>101</sch:EntranceGroupId>
                    <!-- Long. You can find this ID with findEntranceGroup -->
                <sch:DateTimeScheduleId>120</sch:DateTimeScheduleId>
                    <!-- Long. You can find this ID with findDayTimeSchedule -->
                <sch:DateFrom>2019-01-22T00:00:00</sch:DateFrom>
                <sch:DateUntil>2149-05-29T00:00:00</sch:DateUntil>
            </sch:EntranceGroupAuthorisation>
            <sch:EntranceAuthorisation>
                <sch:Enabled>true</sch:Enabled>
                <sch:EntranceId>2</sch:EntranceId>
                    <!-- Long. You can find this ID with findEntrance -->
                <sch:DateTimeScheduleId>120</sch:DateTimeScheduleId>
                <sch:DateFrom>2019-01-22T00:00:00</sch:DateFrom>
                <sch:DateUntil>2149-05-29T00:00:00</sch:DateUntil>
            </sch:EntranceAuthorisation>
        </sch:AuthorisationOnline>
        <sch:AccessDuringHoliday>true</sch:AccessDuringHoliday>
        </sch:ProfileChange>
</soapenv:Body>
```

*Example code for **changeCarrierProfile**.*

```
<soap:Body>
    <ProfileResult>
        <CarrierId>152</CarrierId>
        <AuthorisationOnline>
            <TemplateAuthorisation>
                <Id>19022</Id>
                <Enabled>true</Enabled>
                <TemplateId>51</TemplateId>
                <DateFrom>2019-01-21T00:00:00</DateFrom>
            </TemplateAuthorisation>
            <EntranceGroupAuthorisation>
                <Id>19021</Id>
                <Enabled>true</Enabled>
                <EntranceGroupId>101</EntranceGroupId>
                <DateTimeScheduleId>120</DateTimeScheduleId>
                <DateFrom>2019-01-22T00:00:00</DateFrom>
            </EntranceGroupAuthorisation>
            <EntranceAuthorisation>
                <Id>19020</Id>
                <Enabled>true</Enabled>
                <EntranceId>2</EntranceId>
                <DateTimeScheduleId>120</DateTimeScheduleId>
                <DateFrom>2019-01-22T00:00:00</DateFrom>
            </EntranceAuthorisation>
        </AuthorisationOnline>
        <AccessDuringHoliday>true</AccessDuringHoliday>
    </ProfileResult>
</soap:Body>
```

*Response for the previous code.*




*Result in AEOS*

### 3.1.5 Delete someone

1. Find the ID of the carrier (3.1.3) that you want to delete.
2. Use the function **removeEmployee** / **removeVisitor** / **removeContractor** / **removeCar**.
   The **EmployeeId** (or **VisitorId** etc.) field is mandatory.

```
<soapenv:Body>
    <sch:EmployeeId>104</sch:EmployeeId>
        <!-- Long. You can find this ID with findEmployee -->
</soapenv:Body>
```

*Example code for **removeEmployee**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <RemoveResult xsi:type="xs:string" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">(null)</RemoveResult>
    </soap:Body>
</soap:Envelope>
```

*Response for the previous code. This response means the removal was a success.*

### 3.1.6 Block someone

This procedure blocks a carrier and all their badges. To block a single badge because it is lost, stolen or defective, see replace a forgotten, lost, stolen, or defective badge (3.2.3) or block a badge (3.2.5) instead.

1. Find the ID of the carrier (3.1.3) that you want to block.
2. To examine if this carrier is already blocked, see find out someone's status (3.1.9).
3. Use the function **blockCarrier**.

   The **CarrierId** and **Reason** fields are mandatory.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.nedap.com/aeosws/schema">
    <soapenv:Header/>
    <soapenv:Body>
        <sch:CarrierIdBlock>
            <sch:CarrierId>103</sch:CarrierId>
                <!-- Long. You can find this ID with findPerson / findCar -->
            <sch:Reason>109</sch:Reason>
                <!-- Long. You can find this ID with findBlockReason -->
        </sch:CarrierIdBlock>
    </soapenv:Body>
</soapenv:Envelope>
```

*Example code for **blockCarrier**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <CarrierStateResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
            <CarrierId>103</CarrierId>
            <States>
                <Blocked>true</Blocked>
                <BlockReason>109</BlockReason>
                <ExcludedFromApb>false</ExcludedFromApb>
                <AutoBlockEnabled>true</AutoBlockEnabled>
                <Special>false</Special>
                <Invisible>false</Invisible>
                <ExcludedFromVerification>false</ExcludedFromVerification>
            </States>
        </CarrierStateResult>
    </soap:Body>
</soap:Envelope>
```

*Response for the previous code. **Blocked** is now **true** for this carrier.*

### 3.1.7 Block someone automatically after a period of inactivity

AEOS can block carriers automatically when they have not used their badge for a specific period of time. The auto block time must be set by the system administrator in the AEOS user interface.

- Use the function **activateAutoBlockOnCarrier**.
- The **CarrierIdSetAb** field is mandatory.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.nedap.com/aeosws/schema">
    <soapenv:Header/>
    <soapenv:Body>
        <sch:CarrierIdSetAB>103</sch:CarrierIdSetAB>
            <!-- Long. You can find this ID with findPerson / findCar -->
    </soapenv:Body>
</soapenv:Envelope>
```

*Example code for **activateAutoBlockOnCarrier**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <CarrierStateResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
            <CarrierId>103</CarrierId>
            <States>
                <Blocked>false</Blocked>
                <ExcludedFromApb>false</ExcludedFromApb>
                <AutoBlockEnabled>true</AutoBlockEnabled>
                <Special>false</Special>
                <Invisible>false</Invisible>
                <ExcludedFromVerification>false</ExcludedFromVerification>
            </States>
        </CarrierStateResult>
    </soap:Body>
</soap:Envelope>
```

*Response for the previous code. **AutoBlockEnabled** is now **true** for this carrier.*

To exclude someone from the auto block function, use **deactivateAutoBlockOnCarrier**.

To examine if someone has auto block activated or not, see find out someone's status (3.1.9).

### 3.1.8 Unblock someone

1. Find the ID of the carrier (3.1.3) that you want to unblock.
2. To examine if this carrier is actually blocked, see find out someone's status (3.1.9).
3. Use the function **unblockCarrier**.
   The **CarrierIdUnBlock** field is mandatory.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.nedap.com/aeosws/schema">
    <soapenv:Header/>
    <soapenv:Body>
        <sch:CarrierIdUnBlock>103</sch:CarrierIdUnBlock>
            <!-- Long. You can find this ID with findPerson / findCar -->
    </soapenv:Body>
</soapenv:Envelope>
```

*Example code for **blockCarrier**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <CarrierStateResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
            <CarrierId>103</CarrierId>
            <States>
                <Blocked>false</Blocked>
                <ExcludedFromApb>false</ExcludedFromApb>
                <AutoBlockEnabled>true</AutoBlockEnabled>
                <Special>false</Special>
                <Invisible>false</Invisible>
                <ExcludedFromVerification>false</ExcludedFromVerification>
            </States>
        </CarrierStateResult>
    </soap:Body>
</soap:Envelope>
```

*Response for the previous code. **Blocked** is now **false** for this carrier.*

All badges of this carrier now work again and the original authorisations are restored.

### 3.1.9 Find out someone's status (blocked, APB, special, etc.)

With this procedure you can find out if someone is blocked (3.1.6), if they use auto block (3.1.7), verification or anti pass back, or if they have the status special or invisible.

See the AEOS terminology overview (6.1) for an explanation of these terms.

- Use the function **findCarrierStates**.
- The **CarrierIdStates** field is mandatory.

```
<soapenv:Body>
    <sch:CarrierIdStates>103</sch:CarrierIdStates>
        <!-- Long. You can find this ID with findPerson / findCar -->
</soapenv:Body>
```

*Example code for **findCarrierStates**.*

```
<soap:Body>
    <CarrierStateResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <CarrierId>103</CarrierId>
        <States>
            <Blocked>true</Blocked>
            <BlockReason>109</BlockReason>
            <ExcludedFromApb>false</ExcludedFromApb>
            <AutoBlockEnabled>true</AutoBlockEnabled>
            <Special>false</Special>
            <Invisible>false</Invisible>
            <ExcludedFromVerification>false</ExcludedFromVerification>
        </States>
    </CarrierStateResult>
</soap:Body>
```

*Response for the previous code.*

## 3.2 Badges (identifiers, tokens)

### 3.2.1 Give someone a badge (assign a token)

This procedure gives a badge to someone who already exists in the system. For people who need to be added first, see add a visitor, employee, contractor, or car to the system (3.1.1).

1. Find the ID of the carrier (3.1.3) that you want to give a badge.
2. Find the identifier type number with the **findIdentifierType** function.
3. Use the function **assignToken**.
   The **CarrierId**, **IdentifierType** and **BadgeNumber** fields are mandatory.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.nedap.com/aeosws/schema">
    <soapenv:Header/>
    <soapenv:Body>
        <sch:IdentifierAdd>
            <sch:CarrierId>152</sch:CarrierId>
                <!-- Long. You can find this ID with findPerson / findCar -->
            <sch:IdentifierType>2</sch:IdentifierType>
                <!-- Long. You can find this ID with findIdentifierType -->
            <sch:BadgeNumber>1AF7</sch:BadgeNumber>
                <!-- String, max. 32 characters -->
        </sch:IdentifierAdd>
    </soapenv:Body>
</soapenv:Envelope>
```

*Example code for **assignToken**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <IdentifierResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
            <Id>101</Id>
            <IdentifierType>2</IdentifierType>
            <BadgeNumber>1AF7</BadgeNumber>
            <Blocked>false</Blocked>
            <Status>1</Status>
                <!-- 0=Free, 1=In use, 2=Blocked, 3=Temporarily blocked,
                     4=In use as replacement -->
        </IdentifierResult>
    </soap:Body>
</soap:Envelope>
```

*Response for the previous code.*

**Maintain person**

**Issue identifiers**

Name **Summers, Jerry MSc**
Person no. **0015**
Department **Engineering**
Unit

**Visible rows: 1**

| | Description ▲ | Identifier | Status | Replaced by | Identifier valid to |
|---|---|---|---|---|---|
| ✕ | Mifare CSN with ID: 1af7 | 1AF7 | in use | | |

*Result in AEOS.*

### 3.2.2 Find out which badges (identifiers, tokens) someone has

With this procedure you can find out all badges and other identifiers that someone has, and the status of these badges.

- Use the function **findCarrierToken**.
- The **CarrierIdToken** field is mandatory.

```
<soapenv:Body>
    <sch:CarrierIdToken>152</sch:CarrierIdToken>
</soapenv:Body>
```

*Example code for **findCarrierToken**.*

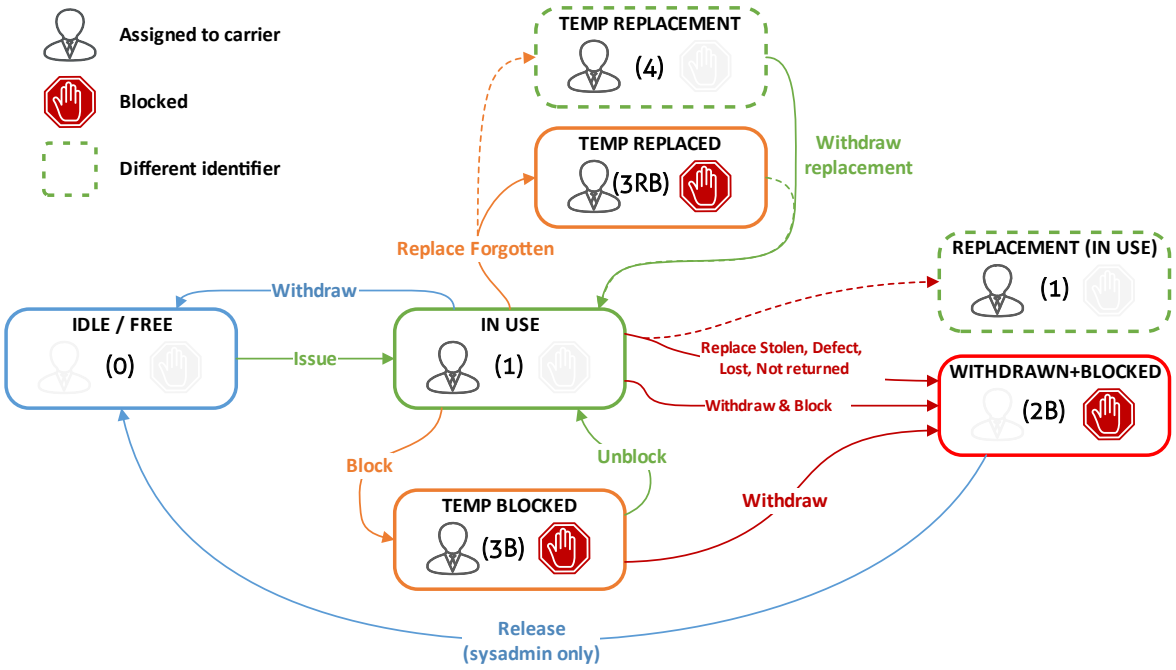
```
<soap:Body>
    <IdentifierList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Identifier>
            <Id>101</Id>
            <IdentifierType>2</IdentifierType>
            <BadgeNumber>1AF7</BadgeNumber>
            <Blocked>false</Blocked>
            <Status>1</Status>
        </Identifier>
    </IdentifierList>
</soap:Body>
```

*Response for the previous code.*

### 3.2.3 Replace a forgotten, lost, stolen, or defective badge

If people have left their badge at home, you can issue a temporary replacement. This blocks the original badge automatically, but keeps it assigned to the one who forgot it. When the replacement badge is later withdrawn, the temporary block on the carrier's original badge will be lifted so that the original badge becomes operational again.

If a badge is lost, stolen, or defective, the original badge will be withdrawn and blocked permanently.




*Identifier states in AEOS.*

The identifier states in SOAP are determined by 3 variables: **Status** (Enum), **Blocked** (Boolean), and **Replaced** (Boolean). The **Status** variable can have these values:
0 = Free, 1 = In use, 2 = Permanently blocked, 3 = Temporarily blocked,
4 = This badge temporarily replaces another badge.

1. Find the ID of the carrier (3.1.3) whose badge you want to replace.
2. Find the badge type and badge number of this carrier's badge (3.2.2).
3. If the replacement badge has a different identifier type (for example, when the replacement badge is not a permanent badge but a printed QR code), you can find the type number with the **findIdentifierType** function.
4. Use the function **replaceToken**.

   Al fields except **EndValidity** are mandatory.

   Enter the (number of the) block reason in the **Reason** field. The block reason that you select determines if this badge is replaced temporarily (status 3RB, reason **Forgot**) or permanently (status 2B, all other reasons).

   If you want to replacement badge to have a limited validity, so that the carrier cannot keep the original badge for too long, define the end date with the **EndValidity** field. If this field is not specified, the replacement badge will stay valid indefinitely, until it is withdrawn manually.

```
<soapenv:Body>
    <sch:IdentifierReplace>
        <sch:CarrierId>152</sch:CarrierId>
            <!-- Long. You can find this ID with findPerson / findCar -->
        <sch:IdentifierTypeFrom>2</sch:IdentifierTypeFrom>
            <!-- Long. You can find this with findCarrierToken -->
        <sch:BadgeNumberFrom>1AF7</sch:BadgeNumberFrom>
            <!-- String, max. 32 characters. You can find this with findCarrierToken -->
        <sch:IdentifierTypeTo>2</sch:IdentifierTypeTo>
            <!-- Long. You can find this ID with findIdentifierType -->
        <sch:BadgeNumberTo>1E28</sch:BadgeNumberTo>
            <!-- String, max. 32 characters. -->
        <sch:Reason>5</sch:Reason> <!-- Forgot (number is different in each system) -->
            <!-- Long. You can find this with findBlockReason -->
        <sch:EndValidity>2019-12-31T23:59:59</sch:EndValidity>
            <!-- Optional. XSD DateTime data type format -->
    </sch:IdentifierReplace>
</soapenv:Body>
```

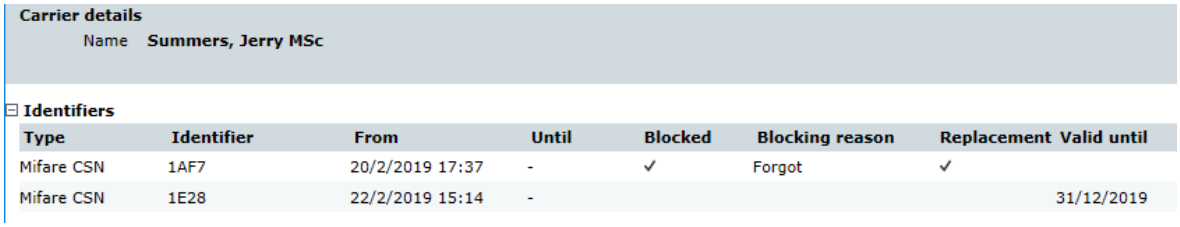
*Example code for **replaceToken**.*

```
<soap:Body>
    <IdentifierResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Id>102</Id>
        <IdentifierType>2</IdentifierType>
        <BadgeNumber>1E28</BadgeNumber>
        <Blocked>false</Blocked>
        <Status>4</Status>
    </IdentifierResult>
</soap:Body>
```

*Response for the previous code.*

```
<soap:Body>
    <IdentifierList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Identifier>
            <Id>101</Id>
            <IdentifierType>2</IdentifierType>
            <BadgeNumber>1AF7</BadgeNumber>
            <Blocked>true</Blocked>
            <Replaced>true</Replaced>
            <ReplacedBy>102</ReplacedBy>
            <DateBlocked>2019-02-22T15:14:01</DateBlocked>
            <BlockReason>5</BlockReason>
            <Status>3</Status>
        </Identifier>
        <Identifier>
            <Id>102</Id>
            <IdentifierType>2</IdentifierType>
            <BadgeNumber>1E28</BadgeNumber>
            <Blocked>false</Blocked>
            <Status>4</Status>
        </Identifier>
    </IdentifierList>
</soap:Body>
```

*Response from findCarrierToken after the replacement, with **Reason** specified as **Forgot**.*



**Carrier details**

Name **Summers, Jerry MSc**

**Identifiers**

| Type | Identifier | From | Until | Blocked | Blocking reason | Replacement | Valid until |
|---|---|---|---|---|---|---|---|
| Mifare CSN | 1AF7 | 20/2/2019 17:37 | - | ✓ | Forgot | ✓ | |
| Mifare CSN | 1E28 | 22/2/2019 15:14 | - | | | | 31/12/2019 |

*Result in AEOS.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <IdentifierList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
            <Identifier>
                <Id>102</Id>
                <IdentifierType>2</IdentifierType>
                <BadgeNumber>1E28</BadgeNumber>
                <Blocked>false</Blocked>
                <Status>1</Status>
            </Identifier>
        </IdentifierList>
    </soap:Body>
</soap:Envelope>
```

*Response from findCarrierToken after the replacement, with **Reason** specified as **Lost**.*
*The new badge is now the only badge of this carrier; the original badge is withdrawn and blocked.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <IdentifierAndCarrierIdList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
            <IdentifierAndCarrierId>
                <Identifier>
                    <Id>101</Id>
                    <IdentifierType>2</IdentifierType>
                    <BadgeNumber>1AF7</BadgeNumber>
                    <Blocked>true</Blocked>
                    <DateBlocked>2019-02-22T17:18:21</DateBlocked>
                    <BlockReason>2</BlockReason>
                    <Status>2</Status>
                </Identifier>
            </IdentifierAndCarrierId>
        </IdentifierAndCarrierIdList>
    </soap:Body>
</soap:Envelope>
```

*Response from findToken for the replaced badge, with **Reason** specified as **Lost**.*
*This badge is now withdrawn and blocked.*

| Identifier ▲ | Carrier... | First name... | ID type | Badge status | Blocked | Blocking reason | Remark | Date/time blocked |
|---|---|---|---|---|---|---|---|---|
| 1AF7 | | | Mifare CSN | | ✓ | Lost | | 22/2/2019 17:18:21 |

*Result in AEOS.*

### 3.2.4 Take back a badge (withdraw a token)

This is typically done when employees who no longer work for the company, or visitors who leave the building, return their badges. Or, when a temporary replacement badge is returned.

If you withdraw a badge, it can be issued to someone else later. If you want to prevent the badge from being used again, for example because it was lost or stolen, block the badge (3.2.5) instead.

1. Find the ID of the carrier (3.1.3) whose badge you want to withdraw.
2. Find the badge type and badge number of this carrier's badge (3.2.2).
   This function also finds if the current active badge is a replacement of another badge, and which badge the original was.
3. Use the function **withdrawToken**.

   The **IdentifierType** and **BadgeNumber** fields are mandatory.
   Set the **RestoreReplaced** field to true if this badge was the replacement for another badge.

```
<soapenv:Body>
    <sch:IdentifierWithdraw>
        <sch:IdentifierType>2</sch:IdentifierType>
            <!-- Long. You can find this with findCarrierToken -->
        <sch:BadgeNumber>1E28</sch:BadgeNumber>
            <!-- String, max. 32 characters. You can find this with findCarrierToken -->
        <sch:RestoreReplaced>true</sch:RestoreReplaced>
    </sch:IdentifierWithdraw>
</soapenv:Body>
```

*Example code for **withdrawToken**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <RemoveResult xsi:type="xs:string" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">(null)</RemoveResult>
    </soap:Body>
</soap:Envelope>
```

*Response for the previous code. This response means the withdrawal was a success.*

4. To check if all badges have been withdrawn and restored correctly, repeat step 2.

### 3.2.5 Block a badge temporarily

You can also block a person instead of a badge, see block someone (3.1.6).

1. Find the ID of the carrier (3.1.3) whose badge you want to block.
2. Find the badge type and badge number of this carrier's badge (3.2.2).
3. Use the function **blockToken**.

   Al fields are mandatory.

   Enter the (number of the) block reason in the **Reason** field. In contrast to the **replaceToken** function, the block reason that you select does not determine if this badge is replaced temporarily or permanently. The badge status will always become 3B: temporarily blocked.

```
<soapenv:Body>
    <sch:IdentifierBlock>
        <sch:IdentifierType>2</sch:IdentifierType>
            <!-- Long. You can find this with findCarrierToken -->
        <sch:BadgeNumber>1E28</sch:BadgeNumber>
            <!-- String, max. 32 characters. You can find this with findCarrierToken -->
        <sch:Reason>5</sch:Reason> <!-- Forgot (number is different in each system) -->
            <!-- Long. You can find this with findBlockReason -->
    </sch:IdentifierBlock>
</soapenv:Body>
```

*Example code for **blockToken**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body/>
</soap:Envelope>
```

*Response for the previous code. This response means the badge was blocked successfully.*

```
<soap:Body>
    <IdentifierAndCarrierIdList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <IdentifierAndCarrierId>
            <Identifier>
                <Id>102</Id>
                <IdentifierType>2</IdentifierType>
                <BadgeNumber>1E28</BadgeNumber>
                <Blocked>true</Blocked>
                <DateBlocked>2019-02-25T10:21:53</DateBlocked>
                <BlockReason>5</BlockReason>
                <Status>3</Status>
            </Identifier>
            <CarrierId>152</CarrierId>
        </IdentifierAndCarrierId>
    </IdentifierAndCarrierIdList>
</soap:Body>
```

*Response from **findToken** for the blocked badge, with **Reason** specified as **Forgot**.*
*This badge is now temporarily blocked, and still assigned to its carrier.*

**Carrier details**

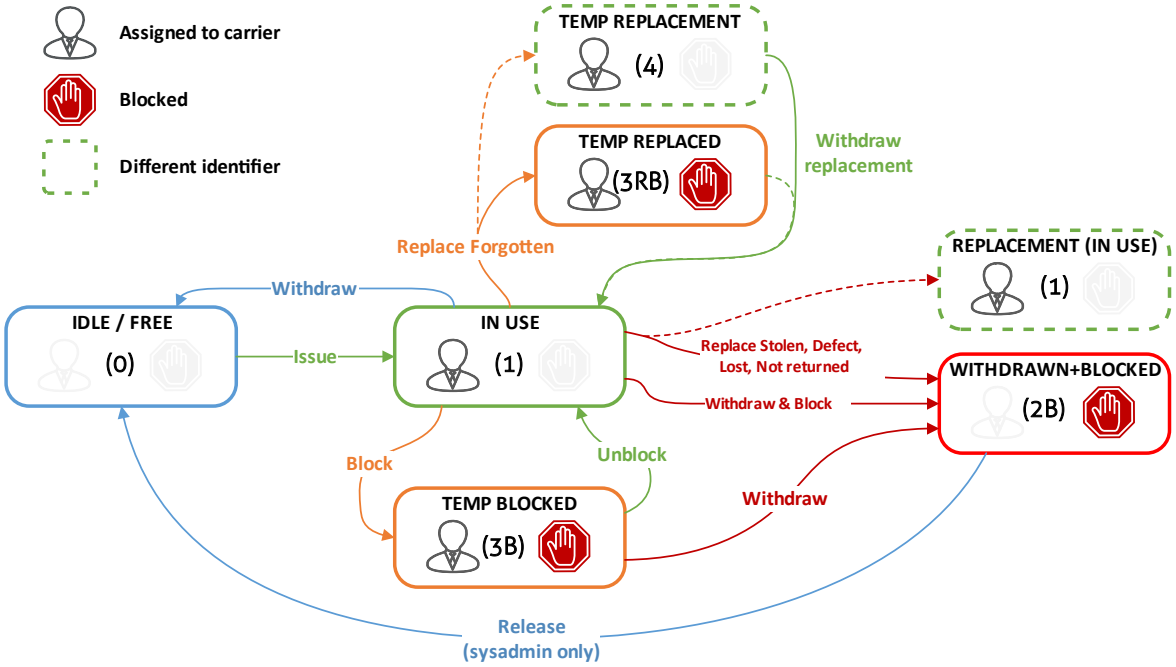
Name **Summers, Jerry MSc**

**Identifiers**

| Type | Identifier | From | Until | Blocked | Blocking reason | Replacement | Valid until |
|---|---|---|---|---|---|---|---|
| Mifare CSN | 1E28 | 22/2/2019 17:18 | - | ✓ | Forgot | | |

*Result in AEOS.*

### 3.2.6 Unblock a badge




*Identifier states in AEOS.*

The identifier states in SOAP are determined by 3 variables: **Status** (Enum), **Blocked** (Boolean), and **Replaced** (Boolean). The **Status** variable can have these values:
0 = Free, 1 = In use, 2 = Permanently blocked, 3 = Temporarily blocked,
4 = This badge temporarily replaces another badge.

How to unblock a badge depends on how it was blocked. A badge can be blocked in several ways:

- Replaced with **replaceToken** and **Forgotten** is selected as **Blocking Reason** (status 3RB). This usually happens when a badge is forgotten at home. In this case, withdraw the replacement badge (3.2.4) to unblock the original badge.
- Blocked temporarily with **blockToken** (status 3B). In this case, unblock the badge (3.2.6.2).
- Blocked with **Withdraw and block identifier**, or it is replaced with **replaceToken** and the **Blocking Reason** is **Defect**, **Lost**, **Not returned** or **Stolen**. In this case, the badge is withdrawn from its carrier and is blocked permanently (status 2B), so it is no longer possible to issue it to anyone else. In this case, a system administrator can release the badge (3.2.6.3) if it has become available again.

#### 3.2.6.1 Unblock a forgotten badge that was replaced

This only works for a badge that has the status *temporarily blocked due to replacement* (status 3RB).

1. Withdraw the replacement badge (3.2.4).
   This automatically unblocks the original badge.

### 3.2.6.2 Unblock a temporarily blocked badge that is assigned to a carrier

This only works for a badge that has been *temporarily blocked* manually (status 3B).

If the badge is *temporarily blocked due to replacement* (status 3RB), withdraw the replacement (3.2.4) to unlock the original badge instead.

1. Find the ID of the carrier (3.1.3) whose badge you want to unblock.
2. Find the badge type and badge number of this carrier's badge (3.2.2).
3. Use the function **unblockToken**.
   Al fields are mandatory.

```
<soapenv:Body>
    <sch:IdentifierUnblock>
        <sch:IdentifierType>2</sch:IdentifierType>
            <!-- Long. You can find this with findCarrierToken -->
        <sch:BadgeNumber>1E28</sch:BadgeNumber>
            <!-- String, max. 32 characters. You can find this with findCarrierToken -->
    </sch:IdentifierUnblock>
</soapenv:Body>
```

*Example code for **unblockToken**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body/>
</soap:Envelope>
```

*Response for the previous code. This response means the badge was unblocked successfully.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <IdentifierAndCarrierIdList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
            <IdentifierAndCarrierId>
                <Identifier>
                    <Id>102</Id>
                    <IdentifierType>2</IdentifierType>
                    <BadgeNumber>1E28</BadgeNumber>
                    <Blocked>false</Blocked>
                    <Status>1</Status>
                </Identifier>
                <CarrierId>152</CarrierId>
            </IdentifierAndCarrierId>
        </IdentifierAndCarrierIdList>
    </soap:Body>
</soap:Envelope>
```

*Response from findToken for the unblocked badge. The badge status is 1 (in use) again.*

**Carrier details**

Name **Summers, Jerry MSc**

**Identifiers**

| Type | Identifier | From | Until | Blocked | Blocking reason | Replacement | Valid until |
|---|---|---|---|---|---|---|---|
| Mifare CSN | 1E28 | 22/2/2019 17:18 | - | ✓ | Forgot | | |

*Result in AEOS.*

### 3.2.6.3 Release a permanently blocked badge that is not assigned to a carrier

If a badge was withdrawn and blocked, the badge became blocked permanently (status 2B). It is no longer possible to issue the badge to anyone. This was usually done for a serious reason, for example, the badge was stolen or permanently lost.

However, a system administrator can still release and unblock the badge if necessary.

1. If you do not know the badge type, you can find it with the **findIdentifierType** function.
2. If you do not know the badge number, you can find it with the **findToken** function. It is not possible to specify **Blocked** as a search criterium, so select the correct **IdentifierType**, and this function will return all badges of this type. You can then filter the response for badges with the **Blocked** status.

   **NOTE:** Searching for all badges can take very long in a large system. If possible, select **UnitID** as additional search criterium. See AEOS terminology overview (6.1) for more information about units.
3. Use the function **releaseBlockedToken**.

   All fields are mandatory.

```
<soapenv:Body>
    <sch:IdentifierRelease>
        <sch:IdentifierType>2</sch:IdentifierType>
            <!-- Long. You can find this with findIdentifierType -->
        <sch:BadgeNumber>1E28</sch:BadgeNumber>
            <!-- String, max. 32 characters. You can find this with findToken -->
    </sch:IdentifierRelease>
</soapenv:Body>
```

*Example code for **releaseBlockedToken**.*

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <RemoveResult xsi:type="xs:string" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">(null)</RemoveResult>
    </soap:Body>
</soap:Envelope>
```

*Response for the previous code. This response means that the badge was released successfully.*

```
<soap:Body>
    <IdentifierAndCarrierIdList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <IdentifierAndCarrierId>
            <Identifier>
                <Id>102</Id>
                <IdentifierType>2</IdentifierType>
                <BadgeNumber>1E28</BadgeNumber>
                <Blocked>false</Blocked>
                <Status>0</Status>
            </Identifier>
        </IdentifierAndCarrierId>
    </IdentifierAndCarrierIdList>
</soap:Body>
```

*Response from findToken for the released badge. The badge status is now 0 (free).*

# 4. Configure an AEOS system with SOAP

This chapter gives an overview of the most common steps to set up AEOS with SOAP.

For information on day-to-day AEOS procedures that you can do with SOAP, see Basic AEOS operations in SOAP (chapter 3) instead.

**To find out what the terms *carrier, access point, entrance* etc. mean, see AEOS terminology (6.1).**

### Settings that can be done with SOAP

1. If necessary, create Entrance Locations Labels (country > site > sub site, see 4.1.1) and/or Entrance Group Labels (region, area or organization, see 4.2.1) that you can assign to Entrances or Entrance Groups when you create them.
2. Create Entrances (4.3.1).
   NOTE: SOAP does not support OSS-SO or offline
3. Confirm all Access Points (4.4.2) and assign them to the created Entrances (4.4.4).
4. If necessary, create Entrance Groups (4.5.1).
5. If necessary, create Count Zones for the Counting function.

   You cannot create Anti Pass Back (APB) zones with SOAP, you can only activate or deactivate APB for specific carriers with SOAP. To create APB zones, use the AEOS user interface instead.
6. Define day/time schedules (4.6.1) and holidays (4.7).
7. Use automatic schedules to automatically unlock public doors during office hours (4.8).
8. Define templates (4.9.1), when and where badges are valid.

**Most code examples do not include the SOAP envelope and header code, which is the same for every call. Make sure to add this code to your own call if you copy the code example directly into you own application. See the add a person example (3.1.1) for a code example where the envelope and header code is included.**

### Settings that must be changed in the AEOS user interface because SOAP does not support these

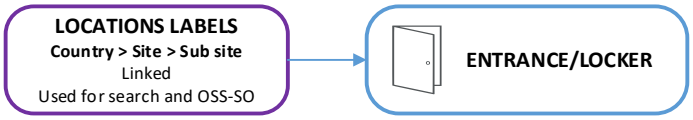
1. Add or change/update OSS-SO (Soaa) or OffLine Entrances or Entrance Groups.
2. Define *Identifier Types* to match the ones that were created in AEmon.
3. Define *Block Reasons* for when carriers or identifiers are blocked.
4. Define the *Auto Block Period*, the time after which someone is blocked due to inactivity.
5. Create an *Entrance Priority List* if you want to make sure that entrances close to the location where badges are issued will have their authorisations updated first.
6. Create *Entrance Filters* if you want AEOS users to see only the Entrances that are close by.

   **NOTE:** This only works if SOAP connects to the AEOS server with a different username/password for every SOAP user.
7. If necessary, create *Emergency Entrance Groups*, groups of entrances that you can lock or unlock with a single operation during an emergency.

See the *AEOS: Create Entrances and Identifier Types* chapter in the *AEOS Access Points and Entrances* manual for more information on these functions.

## 4.1 Entrance locations labels (Country > Site > SubSite)

Locations labels (also called physical entrance labels) can define the exact location of an entrance by **Country**, **Site** and **Sub site**. You can use this later to search more easily for entrances by their location, especially if you work in a large, multi-site organization.




These labels are linked, which means that you need to create a **Country** and all of its **Site**s and **SubSite**s in one call.

There are some AEOS functions that need *locations labels* to be set up correctly.
For example, the OSS-SO updater needs a **Site** ID in its **OSS-SO Site id** field.

### 4.1.1 Create an entrance location label

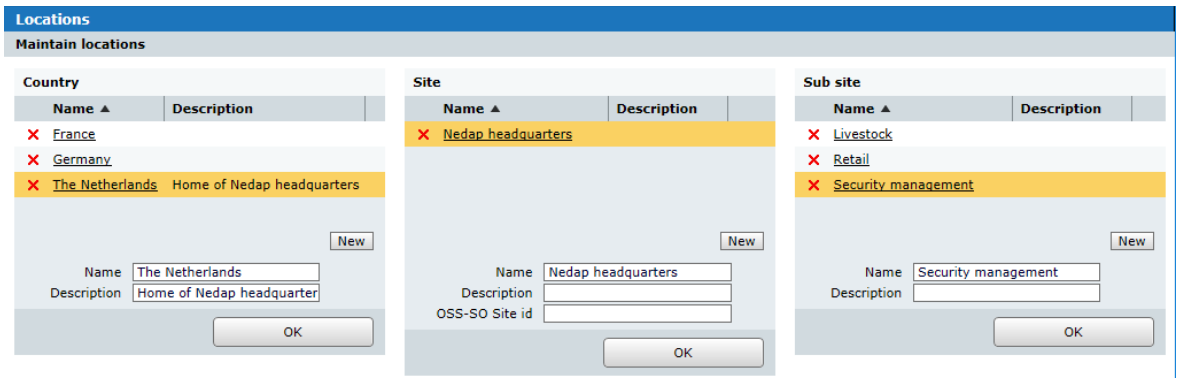
- Use the function **addCountry**.
- The **Name** field is mandatory for the **CountryAdd** section, and for every optional **Site** and **SubSite** section that you use.
- The **ID** fields will be generated by the system.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.nedap.com/aeosws/schema">
    <soapenv:Header/>
    <soapenv:Body>
        <sch:CountryAdd>
            <sch:Name>The Netherlands</sch:Name>
                <!-- String, max. 50 characters -->
            <sch:Description>Home of Nedap headquarters</sch:Description>
                <!-- String, max. 50 characters -->
            <sch:Site>
                <sch:Name>Nedap headquarters</sch:Name>
                <sch:SubSite>
                    <sch:Name>Security management</sch:Name>
                </sch:SubSite>
                <sch:SubSite>
                    <sch:Name>Retail</sch:Name>
                </sch:SubSite>
                <sch:SubSite>
                    <sch:Name>Livestock</sch:Name>
                </sch:SubSite>
            </sch:Site>
        </sch:CountryAdd>
    </soapenv:Body>
</soapenv:Envelope>
```

*Example code for **addCountry**.*

```
<soap:Body>
    <CountryResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Id>51</Id>
        <Name>The Netherlands</Name>
        <Description>Home of Nedap headquarters</Description>
        <Site>
            <Id>52</Id>
            <Name>Nedap headquarters</Name>
            <SubSite>
                <Id>53</Id>
                <Name>Security management</Name>
            </SubSite>
            <SubSite>
                <Id>54</Id>
                <Name>Retail</Name>
            </SubSite>
            <SubSite>
                <Id>55</Id>
                <Name>Livestock</Name>
            </SubSite>
        </Site>
    </CountryResult>
</soap:Body>
```

*Response for the previous code.*




*Result in AEOS.*

**NOTE:** The AEOS **OSS-SO Site ID** field cannot be added with SOAP.

### 4.1.2 Find an entrance location label

- Use the function **findCountry**.
- If you expect that this call returns a large number of items, you can specify the maximum amount of records to return (2.5.5).

```
<soapenv:Body>
    <sch:CountrySearchInfo>
    </sch:CountrySearchInfo>
</soapenv:Body>
```

*Example code for **findCountry**.*

```
<soap:Body>
    <CountryList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Country>
            <Id>2</Id>
            <Name>France</Name>
        </Country>
        <Country>
            <Id>8</Id>
            <Name>Germany</Name>
        </Country>
        <Country>
            <Id>51</Id>
            <Name>The Netherlands</Name>
            <Description>Home of Nedap headquarters</Description>
            <Site>
                <Id>52</Id>
                <Name>Nedap headquarters</Name>
                <SubSite>
                    <Id>53</Id>
                    <Name>Security management</Name>
                </SubSite>
                <SubSite>
                    <Id>54</Id>
                    <Name>Retail</Name>
                </SubSite>
                <SubSite>
                    <Id>55</Id>
                    <Name>Livestock</Name>
                </SubSite>
            </Site>
        </Country>
    </CountryList>
</soap:Body>
```

*Response for the previous code.*

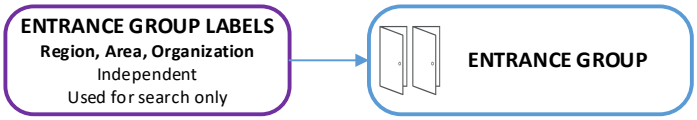
### 4.1.3 Add an entrance location label to an entrance

You can add an entrance location label to an entrance when you create an entrance (4.3.1) or when you change an entrance (4.3.3).

You only need to assign the **SubSiteId**, because this automatically defines the **Site** and **Country**.

## 4.2 Entrance group labels (Region, Area and Organization)

Entrance group labels can define the location of an entrance group by **Region**, **Area** and **Organization**. You can use this later to search more easily for entrance groups by their location, especially if you work in a large, multi-site organization.




**These labels are independent (not linked), which means that you can create Region, Area and Organization labels in any order. You do not have to create all types, and you do not have to create one type before you can create another.**

### 4.2.1 Create an entrance group label

- Use the function **addRegion**, **addArea**, or **addOrganization**.
- The **Name** field is mandatory in each of these functions.
- The **ID** fields will be generated by the system.

```
<soapenv:Body>
    <sch:RegionAdd>
        <sch:Name>AMEC</sch:Name>
            <!-- String, max. 50 characters -->
        <sch:Description>Americas</sch:Description>
            <!-- String, max. 50 characters -->
    </sch:RegionAdd>
</soapenv:Body>
```

*Example code for **addRegion**.*

```
<soapenv:Body>
    <sch:AreaAdd>
        <sch:Name>U.S. East</sch:Name>
            <!-- String, max. 50 characters -->
        <sch:Description>Incl. Michigan to Mississippi</sch:Description>
            <!-- String, max. 50 characters -->
    </sch:AreaAdd>
</soapenv:Body>
```

*Example code for **addArea**.*

```
<soapenv:Body>
    <sch:OrganizationAdd>
        <sch:Name>Nedap USA East</sch:Name>
            <!-- String, max. 50 characters -->
        <sch:Description>NYC office</sch:Description>
            <!-- String, max. 50 characters -->
    </sch:OrganizationAdd>
</soapenv:Body>
```

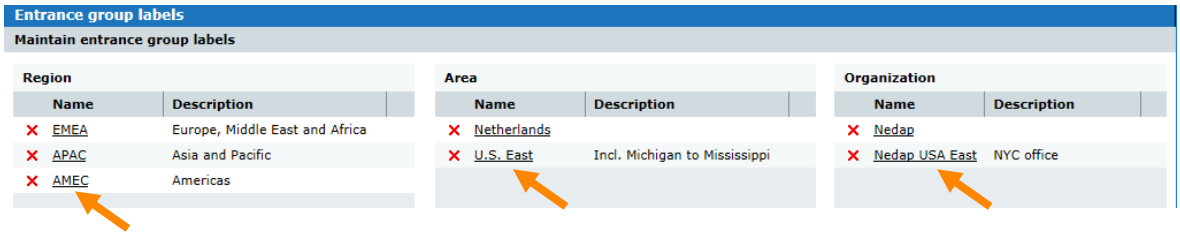
*Example code for **addOrganization**.*

```
<soap:Body>
    <RegionResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Id>101</Id>
        <Name>AMEC</Name>
        <Description>Americas</Description>
    </RegionResult>
</soap:Body>
```

```
<soap:Body>
    <AreaResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Id>51</Id>
        <Name>U.S. East</Name>
        <Description>Incl. Michigan to Mississippi</Description>
    </AreaResult>
</soap:Body>
```

```
<soap:Body>
    <OrganizationResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Id>51</Id>
        <Name>Nedap USA East</Name>
        <Description>NYC office</Description>
    </OrganizationResult>
</soap:Body>
```

*Responses for the previous code snippets.*




*Result in AEOS.*

## 4.2.2 Find an entrance group label

- Use the function **findRegion**, **findArea**, or **findOrganization**.
- If you expect that this call returns a large number of items, you can specify the maximum amount of records to return (2.5.5).

```
<soapenv:Body>
    <sch:RegionSearchInfo>
    </sch:RegionSearchInfo>
</soapenv:Body>
```

*Example code for **findRegion**.*

```
<soapenv:Body>
    <sch:AreaSearchInfo>
    </sch:AreaSearchInfo>
</soapenv:Body>
```

*Example code for **findArea**.*

```
<soapenv:Body>
    <sch:OrganizationSearchInfo>
    </sch:OrganizationSearchInfo>
</soapenv:Body>
```

*Example code for **findOrganization**.*

```
<soap:Body>
    <RegionList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Region>
            <Id>51</Id>
            <Name>AMEC</Name>
            <Description>Americas</Description>
        </Region>
    </RegionList>
</soap:Body>
```

```
<soap:Body>
    <AreaList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Area>
            <Id>51</Id>
            <Name>U.S. East</Name>
            <Description>Incl. Michigan to Mississippi</Description>
        </Area>
    </AreaList>
</soap:Body>
```

```
<soap:Body>
    <OrganizationList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Organization>
            <Id>51</Id>
            <Name>Nedap USA East</Name>
            <Description>NYC office</Description>
        </Organization>
    </OrganizationList>
</soap:Body>
```

*Responses for the previous code snippets.*

### 4.2.3 Add an entrance group label to an entrance group

You can add an entrance group label to an entrance group when you create an entrance group (4.5.1) or when you change an entrance group (4.5.3).

## 4.3 Entrances

### 4.3.1 Create an entrance

- Use the function **addEntrance**.
- The **Name** and **UnitOfAuthType** fields are mandatory.

  **NOTE:** Only *UnitOfAuthType* **OnLine** is supported for this function. SOAP does not support creating/changing OSS-SO (Soaa) or OffLine Entrances or Entrance Groups.
- The **ID** field will be generated by the system.
- For the **SubSiteId** field, see entrance locations labels (4.1).
  For the **scheduleid** fields, see automatic schedules (4.8).

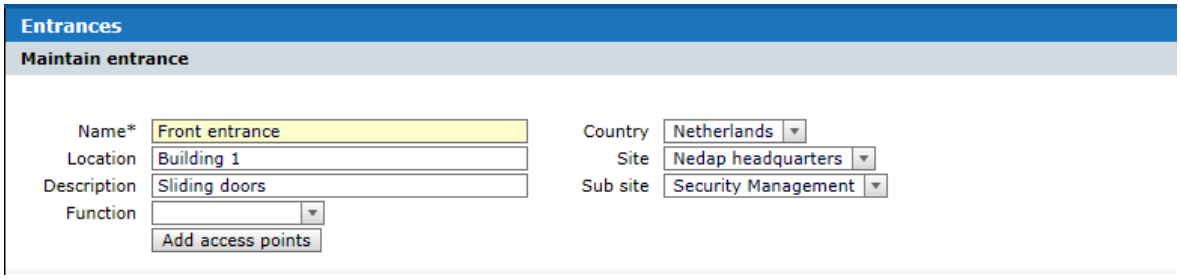
See the AEOS terminology overview (6.1) for additional explanations of Online, Soaa (OSS-SO) and Offline doors, automatic unlock schedules, etc.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.nedap.com/aeosws/schema">
    <soapenv:Header/>
    <soapenv:Body>
        <sch:EntranceAdd>
            <sch:Name>Front entrance</sch:Name>
                <!-- String, max. 150 characters -->
            <sch:Location>Building 1</sch:Location>
                <!-- String, max. 50 characters -->
            <sch:Description>Sliding doors</sch:Description>
                <!-- String, max. 150 characters -->
            <sch:UnitOfAuthType>OnLine</sch:UnitOfAuthType>
                <!-- OnLine only, OffLine and Soaa not supported -->
            <sch:SubSiteId>5</sch:SubSiteId>
                <!-- Long. You can find this ID with findCountry -->
                <!-- The SubSite ID automatically defines the Site and Country -->
            <sch:verificationdisabledschedid>120</sch:verificationdisabledschedid>
            <sch:autounlockscheduleid>120</sch:autounlockscheduleid>
            <sch:autolockscheduleid>169</sch:autolockscheduleid>
            <sch:togglescheduleid>120</sch:togglescheduleid>
                <!-- Long. You can find this ID with findDayTimeSchedule -->
        </sch:EntranceAdd>
    </soapenv:Body>
</soapenv:Envelope>
```

*Example code for **addEntrance.***

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <EntranceResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
            <Id>151</Id>
            <Name>Front entrance</Name>
            <Location>Building 1</Location>
            <Description>Sliding doors</Description>
            <UnitOfAuthType>OnLine</UnitOfAuthType>
            <SubSiteId>5</SubSiteId>
            <verificationdisabledschedid>120</verificationdisabledschedid>
            <autounlockscheduleid>120</autounlockscheduleid>
            <autolockscheduleid>169</autolockscheduleid>
            <togglescheduleid>120</togglescheduleid>
        </EntranceResult>
    </soap:Body>
</soap:Envelope>
```

*Response for the previous code.*




| Name ▲ | Location | Description | Automatic lock... | Automatic unlock... | Toggle schedule | Verification disable |
|---|---|---|---|---|---|---|
| Front entrance | Building 1 | Sliding doors | After hours | Working hours | Working hours | Working hours |

*Result in AEOS.*

**NOTE:** The AEOS **Function** field for entrances is not supported in SOAP.

### 4.3.2 Find an entrance

You can use this to find the ID of an entrance, which you need to change the entrance or add it to a template etc.

- Use the function **findEntrance**.
- The **UnitOfAuthType** field is mandatory, all other fields are optional.
- If you expect that this call returns a large number of items, you can specify the maximum amount of records to return (2.5.5).

```
<soapenv:Body>
   <sch:EntranceSearchInfo>
       <sch:EntranceInfo>
           <sch:UnitOfAuthType>OnLine</sch:UnitOfAuthType>
               <!-- OnLine, OffLine, Soaa -->
       </sch:EntranceInfo>
   </sch:EntranceSearchInfo>
</soapenv:Body>
```

*Example code for **findEntrance**. This code finds all online entrances.*

```
<EntranceList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
    <Entrance>
        <Id>1</Id>
        <Name>Main Entrance</Name>
        <Location>Lobby</Location>
        <Description>Two sliding doors</Description>
        <UnitOfAuthType>OnLine</UnitOfAuthType>
    </Entrance>
    <Entrance>
        <Id>2</Id>
        <Name>Office door</Name>
        <Location>Between Lobby and office</Location>
        <UnitOfAuthType>OnLine</UnitOfAuthType>
    </Entrance>
</EntranceList>
```

*Response for the previous code.*

### 4.3.3 Change an entrance

- Use the function **changeEntrance**.
- The **Id** and **UnitOfAuthType** fields are mandatory.

  **NOTE:** Only *UnitOfAuthType* **OnLine** is supported for this function. SOAP does not support creating/changing OSS-SO (Soaa) or OffLine Entrances or Entrance Groups.
- All other fields are optional. Only specify the fields that you want to add or change.
  Any existing optional fields that are not specified in this call, are not deleted or changed.
- For the **SubSiteId** field, see entrance locations labels (4.1).
  For the **scheduleid** fields, see automatic schedules (4.8).

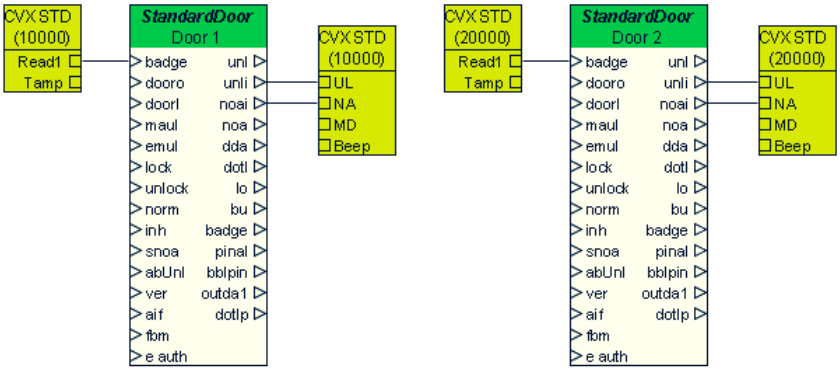
See the AEOS terminology overview (6.1) for additional explanations of Online, Soaa (OSS-SO) and Offline doors, automatic unlock schedules, etc.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.nedap.com/aeosws/schema">
    <soapenv:Header/>
    <soapenv:Body>
        <sch:EntranceChange>
            <sch:Id>151</sch:Id>
                <!-- MANDATORY. Use the value from findEntrance -->
            <sch:Name>Front entrance</sch:Name>
                <!-- String, max. 150 characters -->
            <sch:Location>Building 1</sch:Location>
                <!-- String, max. 50 characters -->
            <sch:Description>Sliding doors</sch:Description>
                <!-- String, max. 150 characters -->
            <sch:UnitOfAuthType>OnLine</sch:UnitOfAuthType>
                <!-- MANDATORY. OnLine only, OffLine and Soaa not supported -->
                <!-- If findEntrance gives OffLine or Soaa, you can't use this. -->
            <sch:SubSiteId>5</sch:SubSiteId>
                <!-- Long. You can find this ID with findCountry -->
                <!-- The SubSite ID automatically defines the Site and Country -->
            <sch:verificationdisabledschedid>120</sch:verificationdisabledschedid>
            <sch:autounlockscheduleid>120</sch:autounlockscheduleid>
            <sch:autolockscheduleid>169</sch:autolockscheduleid>
            <sch:togglescheduleid>120</sch:togglescheduleid>
                <!-- For any of the above 4, use the value from findDayTimeSchedule -->
      </sch:EntranceChange>
   </soapenv:Body>
</soapenv:Envelope>
```

*Example code for **changeEntrance**.*

## 4.4 Access Points

After Access Points are created in the door controllers (AEpus), they let AEOS on the AEserver know that they exist, so that AEOS can find them. Before you can use the new Access Points in AEOS, you need to confirm them first. After that you can add them to an entrance.




*Access Points **Door 1** and **Door 2** on a door controller (AEpu).*

**Confirm access points**

**Confirm automatically detected access points**

**Visible rows: 2**

| Access point ▲ | Host name | Type |
|---|---|---|
| Door 1 | serverroom2-aepu3 | StandardDoor |
| Door 2 | serverroom2-aepu3 | StandardDoor |

*The unconfirmed Access Points as they appear in AEOS.*

### 4.4.1 Find unconfirmed Access Points

- Use the function **findUnconfirmedAccessPoint**.
- There are no mandatory fields. However, you can specify some fields as search criteria.
- If you expect that this call returns a large number of items, you can specify the maximum amount of records to return (2.5.5).

```
<soapenv:Body>
    <sch:UnconfirmedAccessPointSearchInfo>
        <sch:AccessPointInfo>
            <sch:Name>Door</sch:Name>
            <sch:HostName>server</sch:HostName>
            <sch:Type>StandardDoor</sch:Type>
                <!-- Above 3: String, max. 50 characters -->
            <sch:ServiceKey>server</sch:ServiceKey>
                <!-- String, max. 256 characters -->
         </sch:AccessPointInfo>
    </sch:UnconfirmedAccessPointSearchInfo>
</soapenv:Body>
```

*Example code for **findUnconfirmedAccessPoint.***

```
<soap:Body>
    <UnconfirmedAccessPointList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <UnconfirmedAccessPoint>
            <Name>Door 1</Name>
            <HostName>serverroom2-aepu3</HostName>
            <Type>StandardDoor</Type>
            <ServiceKey>serverroom2-aepu3.door 12052710495</ServiceKey>
        </UnconfirmedAccessPoint>
        <UnconfirmedAccessPoint>
            <Name>Door 2</Name>
            <HostName>serverroom2-aepu3</HostName>
            <Type>StandardDoor</Type>
            <ServiceKey>serverroom2-aepu3.door 22052710496</ServiceKey>
        </UnconfirmedAccessPoint>
    </UnconfirmedAccessPointList>
</soap:Body>
```

*Response for the previous code.*

### 4.4.2 Confirm all unconfirmed Access Points

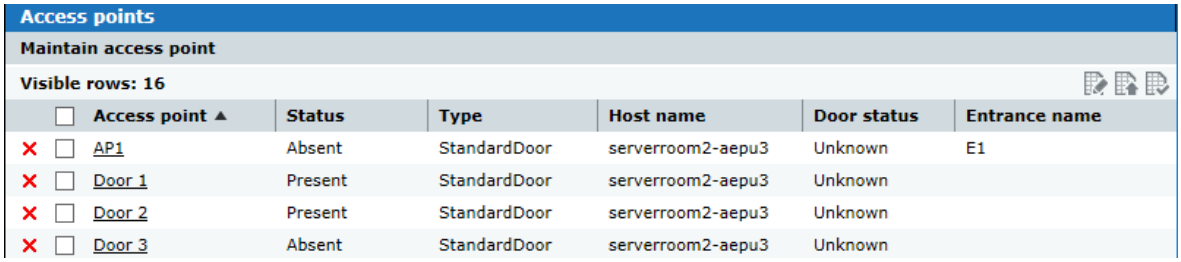
- Use the function **confirmAccessPoints**.
- There are no mandatory fields. However, you can specify some fields as search criteria (see findUnconfirmedAccessPoint, 4.4.1). If you do not specify any search criteria, this function will confirm all unconfirmed Access Points in the system.
- If you expect that this call returns a large number of items, you can specify the maximum amount of records to return (2.5.5).

```
<soapenv:Body>
    <sch:UnconfirmedAccessPointSearchInfo2>
        <sch:AccessPointInfo>
        </sch:AccessPointInfo>
    </sch:UnconfirmedAccessPointSearchInfo2>
</soapenv:Body>
```

*Example code for **confirmAccessPoints**. This code confirms all unconfirmed Access Points in the system.*

```
<soap:Body>
    <AccessPointList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <AccessPoint>
            <Name>Door 1</Name>
            <HostName>serverroom2-aepu3</HostName>
            <Type>StandardDoor</Type>
            <ServiceKey>serverroom2-aepu3.door 12052710495</ServiceKey>
            <Id>155</Id>
        </AccessPoint>
        <AccessPoint>
            <Name>Door 2</Name>
            <HostName>serverroom2-aepu3</HostName>
            <Type>StandardDoor</Type>
            <ServiceKey>serverroom2-aepu3.door 22052710496</ServiceKey>
            <Id>156</Id>
        </AccessPoint>
    </AccessPointList>
</soap:Body>
```

*Response for the previous code.*



Access points

Maintain access point

Visible rows: 16

| | Access point ▲ | Status | Type | Host name | Door status | Entrance name |
|---|---|---|---|---|---|---|
| ✕ | AP1 | Absent | StandardDoor | serverroom2-aepu3 | Unknown | E1 |
| ✕ | Door 1 | Present | StandardDoor | serverroom2-aepu3 | Unknown | |
| ✕ | Door 2 | Present | StandardDoor | serverroom2-aepu3 | Unknown | |
| ✕ | Door 3 | Absent | StandardDoor | serverroom2-aepu3 | Unknown | |

*The confirmed Access Points as they appear in the Access Points list in AEOS.*

### 4.4.3 Find access points

- Use the function **findAccessPoint**.
- There are no mandatory fields.
- If you expect that this call returns a large number of items, you can specify the maximum amount of records to return (2.5.5).

```
<soapenv:Body>
    <sch: AccessPointSearchInfo>
        <sch:AccessPointInfo>
        </sch:AccessPointInfo>
    </sch:AccessPointSearchInfo>
</soapenv:Body>
```

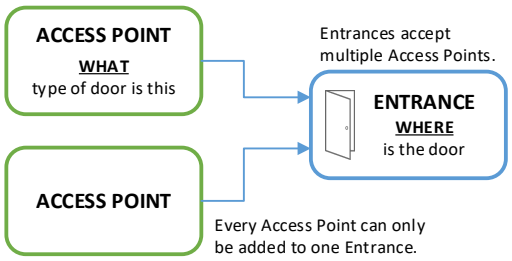
*Example code for **findAccessPoint**. This code finds all confirmed Access Points in the system.*

```
<AccessPointList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
    <AccessPoint>
        <Name>AP1</Name>
        <HostName>serverroom2-aepu3</HostName>
        <Type>StandardDoor</Type>
        <ServiceKey>serverroom2-aepu3.ap164994</ServiceKey>
        <Id>7</Id>
        <EntranceId>5</EntranceId>
    </AccessPoint>
    <AccessPoint>
        <Name>Door 1</Name>
        <HostName>serverroom2-aepu3</HostName>
        <Type>StandardDoor</Type>
        <ServiceKey>serverroom2-aepu3.door 12052710495</ServiceKey>
        <Id>155</Id>
    </AccessPoint>
    <AccessPoint>
        <Name>Door 2</Name>
        <HostName>serverroom2-aepu3</HostName>
        <Type>StandardDoor</Type>
        <ServiceKey>serverroom2-aepu3.door 22052710496</ServiceKey>
        <Id>156</Id>
    </AccessPoint>
    <AccessPoint>
        <Name>Door 3</Name>
        <HostName>serverroom2-aepu3</HostName>
        <Type>StandardDoor</Type>
        <ServiceKey>serverroom2-aepu3.door 32052710497</ServiceKey>
        <Id>106</Id>
    </AccessPoint>
</AccessPointList>
```

*Response for the previous code.*

### 4.4.4 Add an access point to an entrance

You need to add Access Points to an Entrance, to control the authorisations of those Access Points.




1. Find the entrance ID (4.3.2) of the entrance that you want to add the access point to.
   This entrance must have the **UnitOfAuthType** value **OnLine** , because you can only add access points to online entrances.
2. Find the Access Point ID (4.4.3) of the access point that you want to add to this entrance.
3. Use the function **changeAccessPoint**.
   The **Id** field is mandatory. All other fields are optional.
   Use the **EntranceId** field to assign this Access Point to an Entrance.

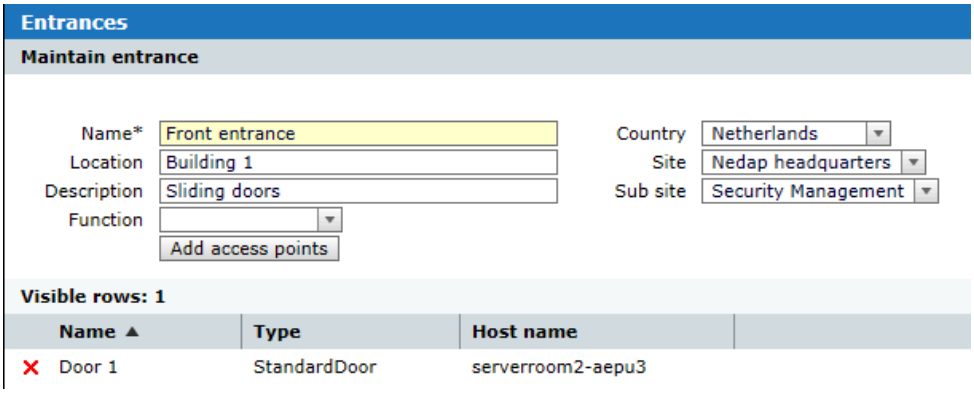
See the AEOS terminology overview (6.1) for additional explanations of access points and entrances.

```
<soapenv:Body>
    <sch:AccessPointChange>
        <sch:Id>156</sch:Id>
            <!-- MANDATORY. Use the value from findAccessPoint -->
        <sch:Description>Door no. 1</sch:Description>
            <!-- String, max. 50 characters -->
        <sch:EntranceId>151</sch:EntranceId>
            <!-- Long. You can find this ID with findEntrance -->
    </sch:AccessPointChange>
</soapenv:Body>
```

*Example code for **changeAccessPoint**. This code assigns Access Point 156 to Entrance 151.*

```
<soap:Body>
    <AccessPointResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Name>Door 1</Name>
        <HostName>serverroom2-aepu3</HostName>
        <Type>StandardDoor</Type>
        <ServiceKey>serverroom2-aepu3.door 12052710495</ServiceKey>
        <Id>156</Id>
        <Description>Door no. 1</Description>
        <EntranceId>151</EntranceId>
    </AccessPointResult>
</soap:Body>
```

*Response for the previous code.*

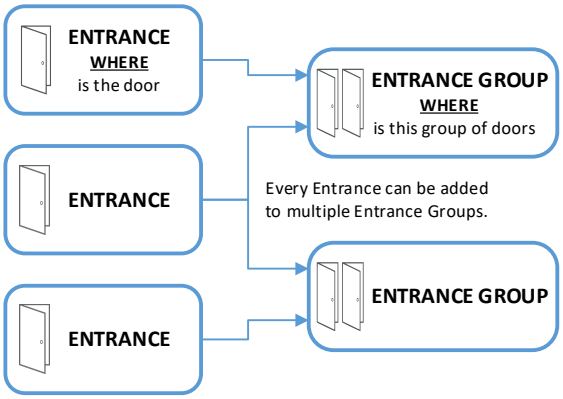



*Result in AEOS: the Access Point has been added to Entrance 151 from create an entrance (4.3.1).*

# 4.5 Entrance groups

## 4.5.1 Create an entrance group

Entrances that are not close together, but still have the same authorisation, can be grouped into Entrance Groups. For example, all emergency exits, or all of the doors in a single office unit. This way, Entrance Groups make it easier to give people access to a large number of Entrances.




**You can only add entrances to entrance groups of their own type (on line, offline, OSS-SO etc.). However, SOAP does not support creating/changing OSS-SO (Soaa) or OffLine Entrances or Entrance groups.**

- Use the function **addEntranceGroup**.
- The **Name** and **UnitOfAuthType** fields are mandatory.

  **NOTE:** Only *UnitOfAuthType* values of **OnLine**, **SecurityLevel**, and **Loxs** are supported for this function. SOAP does not support creating/changing OSS-SO (Soaa) or OffLine Entrances or Entrance Groups.
- The **ID** field will be generated by the system.
- For the **RegionId** , **AreaId** and **OrganizationId** fields, see entrance group labels (4.2).
  For the **SiteId** field, see entrance locations labels (4.1).
  For the **EntranceIdDirection** fields, see Security Level Management.

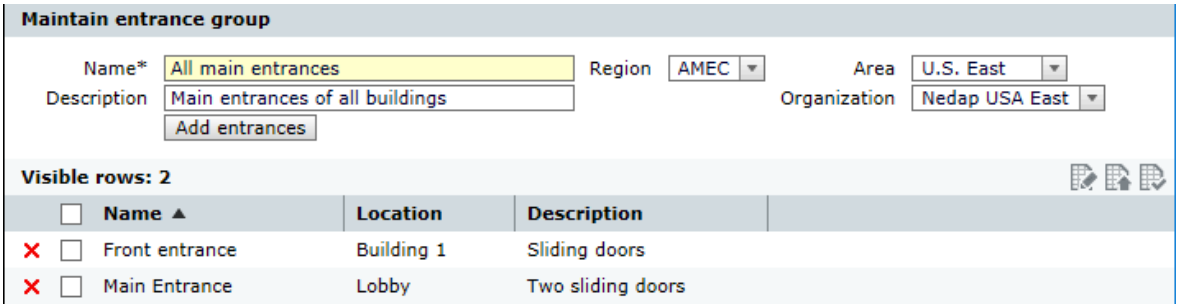
See the AEOS terminology overview (6.1) for additional explanations of Online, Soaa (OSS-SO) and Offline doors, Security Level Management, etc.

```
<soapenv:Body>
    <sch:EntranceGroupAdd>
        <sch:Name>All main entrances</sch:Name>
            <!-- String, max. 150 characters -->
        <sch:Description>Main entrances of all buildings</sch:Description>
            <!-- String, max. 150 characters -->
        <sch:UnitOfAuthType>OnLine</sch:UnitOfAuthType>
            <!-- OnLine, SecurityLevel, Loxs -->
        <sch:EntranceIdList>
            <sch:EntranceId>1</sch:EntranceId>
            <sch:EntranceId>151</sch:EntranceId>
                <!-- Long. You can find this ID with findEntrance -->
        </sch:EntranceIdList>
        <sch:RegionId>101</sch:RegionId>
        <sch:AreaId>51</sch:AreaId>
        <sch:OrganizationId>51</sch:OrganizationId>
            <!-- Long. Find these with findRegion, findArea, and findOrganization -->
    </sch:EntranceGroupAdd>
</soapenv:Body>
```

*Example code for **addEntranceGroup**.*

```
<soap:Body>
    <EntranceGroupResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Id>101</Id>
        <Name>All main entrances</Name>
        <Description>Main entrances of all buildings</Description>
        <UnitOfAuthType>OnLine</UnitOfAuthType>
        <EntranceIdList>
            <EntranceId>151</EntranceId>
            <EntranceId>1</EntranceId>
        </EntranceIdList>
        <RegionId>101</RegionId>
        <AreaId>51</AreaId>
        <OrganizationId>51</OrganizationId>
    </EntranceGroupResult>
</soap:Body>
```

*Response for the previous code.*




*Result in AEOS.*

### 4.5.2 Find an entrance group

You can use this to find the ID of an entrance group, which you need to change the entrance group or add it to a template etc.

- Use the function **findEntranceGroup**.
- The **UnitOfAuthType** field is mandatory, all other fields are optional.
- If you expect that this call returns a large number of items, you can specify the maximum amount of records to return (2.5.5).

```
<soapenv:Body>
    <sch:EntranceGroupSearchInfo>
        <sch:EntranceGroupInfo>
            <sch:UnitOfAuthType>OnLine</sch:UnitOfAuthType>
                <!-- OnLine, OffLine, SecurityLevel, Soaa, Loxs -->
        </sch:EntranceGroupInfo>
    </sch:EntranceGroupSearchInfo>
</soapenv:Body>
```

*Example code for **findEntranceGroup**. This code finds all online entrance groups.*

```
<soap:Body>
    <EntranceGroupList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <EntranceGroup>
            <Id>101</Id>
            <Name>All main entrances</Name>
            <Description>Main entrances of all buildings</Description>
            <UnitOfAuthType>OnLine</UnitOfAuthType>
            <EntranceIdList>
                <EntranceId>151</EntranceId>
                <EntranceId>1</EntranceId>
            </EntranceIdList>
            <RegionId>101</RegionId>
            <AreaId>51</AreaId>
            <OrganizationId>51</OrganizationId>
        </EntranceGroup>
    </EntranceGroupList>
</soap:Body>
```

*Response for the previous code.*

### 4.5.3 Change an entrance group

- Use the function **changeEntranceGroup**.
- The **Id** and **UnitOfAuthType** fields are mandatory.

  **NOTE:** Only *UnitOfAuthType* values of **OnLine**, **SecurityLevel**, and **Loxs** are supported for this function. SOAP does not support creating/changing OSS-SO (Soaa) or OffLine Entrances or Entrance Groups.
- All other fields are optional. Only specify the fields that you want to add or change.
  Any existing optional fields that are not specified in this call, are not deleted or changed.
- If you want to change the **EntranceIdList**, specify the whole list.
  Any existing **EntranceId** entries are overwritten with the new list.
- For the **RegionId** , **AreaId** and **OrganizationId** fields, see entrance group labels (4.2).
  For the **SiteId** field, see entrance locations labels (4.1)and OSS-SO (Soaa in SOAP).
  For the **EntranceIdDirection** fields, see Security Level Management.

See the AEOS terminology overview (6.1) for additional explanations of Online, Soaa (OSS-SO) and Offline doors, Security Level Management, etc.

```
<soapenv:Body>
    <sch:EntranceGroupChange>
        <sch:Id>101</sch:Id>
            <!-- MANDATORY. Use the value from findEntranceGroup -->
        <sch:Name>All main entrances</sch:Name>
            <!-- String, max. 150 characters -->
        <sch:Description>Main entrances of all buildings</sch:Description>
            <!-- String, max. 150 characters -->
        <sch:UnitOfAuthType>OnLine</sch:UnitOfAuthType>
            <!-- MANDATORY. Use the value from findEntranceGroup -->
        <sch:EntranceIdList>
            <sch:EntranceId>1</sch:EntranceId>
            <sch:EntranceId>151</sch:EntranceId>
                <!-- Long. You can find this ID with findEntrance -->
        </sch:EntranceIdList>
        <sch:RegionId>101</sch:RegionId>
        <sch:AreaId>51</sch:AreaId>
        <sch:OrganizationId>51</sch:OrganizationId>
            <!-- Long. Find these with findRegion, findArea, and findOrganization -->
    </sch:EntranceGroupChange>
</soapenv:Body>
```

*Example code for **changeEntranceGroup**.*

## 4.6 Day/time schedules

In AEOS, you can define these 2 types of day/time schedules:

- **Weekly**: repeats every 7 days.
- **Free period**: repeats in any number of 1-365 days.

If your system runs in different time zones, AEOS uses the local time for day/time schedules.

### 4.6.1 Create a day/time schedule

- Use the function **addDayTimeSchedule**.
- The **Name** field is mandatory.
- The **ID** field will be generated by the system.
- The **Restricted** field is optional and turns OSS-SO restrictions on or off for this schedule.

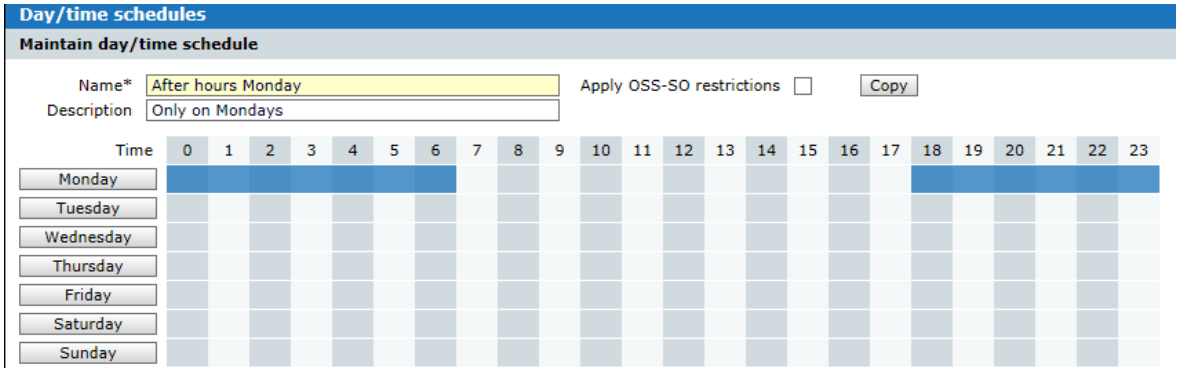
See the AEOS terminology overview (6.1) for additional explanations of day/time schedules and OSS-SO restrictions.

```
<soapenv:Body>
    <sch:DayTimeScheduleAdd>
        <sch:Name>After hours Monday</sch:Name>
            <!-- String, max. 50 characters -->
        <sch:Description>Only on Mondays</sch:Description>
            <!-- String, max. 50 characters -->
        <sch:Type>Week</sch:Type>
            <!-- Week, Free -->
        <sch:StartDate>2018-01-28</sch:StartDate>
            <!-- XSD Date data type format -->
        <sch:LengthInDays>7</sch:LengthInDays>
            <!-- Int, 1 to 365 -->
        <sch:DayPeriod>
            <sch:StartDay>0</sch:StartDay>
                <!-- Int, 0 to 365 -->
                <!-- For week schedules: 0 = Monday, 6 = Sunday -->
            <sch:StartHour>0</sch:StartHour>
                <!-- Int, 0 to 23 -->
            <sch:StartMinute>0</sch:StartMinute>
                <!-- Int, 0 to 59 -->
            <sch:EndDay>0</sch:EndDay>
            <sch:EndHour>7</sch:EndHour>
            <sch:EndMinute>0</sch:EndMinute>
        </sch:DayPeriod>
        <sch:DayPeriod>
            <sch:StartDay>0</sch:StartDay>
            <sch:StartHour>18</sch:StartHour>
            <sch:StartMinute>0</sch:StartMinute>
            <sch:EndDay>0</sch:EndDay>
            <sch:EndHour>0</sch:EndHour>
            <sch:EndMinute>0</sch:EndMinute>
        </sch:DayPeriod>
        <sch:Restricted>false</sch:Restricted>
    </sch:DayTimeScheduleAdd>
</soapenv:Body>
```

*Example code for **addDayTimeSchedule.***

```
<soap:Body>
    <DayTimeScheduleResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Id>171</Id>
        <Name>After hours Monday</Name>
        <Description>Only on Mondays</Description>
        <Type>Week</Type>
        <StartDate>2002-01-28</StartDate>
        <LengthInDays>7</LengthInDays>
        <DayPeriod>
            <StartDay>0</StartDay>
            <StartHour>0</StartHour>
            <StartMinute>0</StartMinute>
            <EndDay>0</EndDay>
            <EndHour>7</EndHour>
            <EndMinute>0</EndMinute>
        </DayPeriod>
        <DayPeriod>
            <StartDay>0</StartDay>
            <StartHour>18</StartHour>
            <StartMinute>0</StartMinute>
            <EndDay>0</EndDay>
            <EndHour>0</EndHour>
            <EndMinute>0</EndMinute>
        </DayPeriod>
        <Restricted>false</Restricted>
    </DayTimeScheduleResult>
</soap:Body>
```

*Response for the previous code.*




*Result in AEOS.*

## 4.6.2 Find a day/time schedule

You can use this to find the ID of a day/time schedule, which you need to change the schedule or add it to a template etc.

- Use the function **findDayTimeSchedule**.
- If you expect that this call returns a large number of items, you can specify the maximum amount of records to return (2.5.5).

```
 <soapenv:Body>
    <sch:DayTimeScheduleSearchInfo>
    </sch:DayTimeScheduleSearchInfo>
</soapenv:Body>
```

*Example code for **findDayTimeSchedule**.*

```
<soap:Body>
    <DayTimeScheduleList xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <DayTimeSchedule>
            <Id>169</Id>
            <Name>After hours</Name>
            <Description>Night and weekends</Description>
            <Type>Week</Type>
            <StartDate>2002-01-28</StartDate>
            <LengthInDays>7</LengthInDays>
            <DayPeriod>
                <StartDay>0</StartDay>
                <StartHour>0</StartHour>
                <StartMinute>0</StartMinute>
                <EndDay>0</EndDay>
                <EndHour>7</EndHour>
                <EndMinute>0</EndMinute>
            </DayPeriod>
            <DayPeriod>
                <StartDay>0</StartDay>
                <StartHour>18</StartHour>
                <StartMinute>0</StartMinute>
                <EndDay>0</EndDay>
                <EndHour>0</EndHour>
                <EndMinute>0</EndMinute>
            </DayPeriod>
            <DayPeriod>
                <StartDay>1</StartDay>
                <StartHour>0</StartHour>
...
            <DayPeriod>
                <StartDay>6</StartDay>
                <StartHour>0</StartHour>
                <StartMinute>0</StartMinute>
                <EndDay>6</EndDay>
                <EndHour>0</EndHour>
                <EndMinute>0</EndMinute>
            </DayPeriod>
            <Restricted>false</Restricted>
        </DayTimeSchedule>
    </DayTimeScheduleList>
</soap:Body>
```

*Response for the previous code.*

### 4.6.3 Change a day/time schedule

- Use the function **changeDayTimeSchedule**.
- The **Id** field is mandatory.
- All other fields are optional. Only specify the fields that you want to add or change.
  Any existing optional fields that are not specified in this call, are not deleted or changed.

See create a day/time schedule (4.6.1) for all available fields.

## 4.7 Create a holiday period

Entrances that are added to a *holiday* do not open automatically for everyone during office hours (see automatic unlock schedules, 4.8), but they only open for someone with a valid badge.

**You can define different holidays for different countries and areas, by selecting only the entrances in that country or area.**

- Use the function **addHoliday**.
- The **Name**, **ValidFrom** and **ValidUntil** fields are mandatory.
- The **ID** field will be generated by the system.

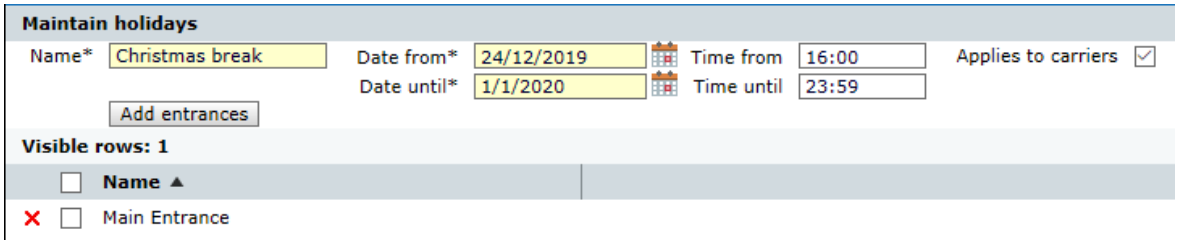
See the AEOS terminology overview (6.1) for an additional explanation of holidays.

```
<soapenv:Body>
    <sch:HolidayAdd>
        <sch:Name>Christmas break</sch:Name>
            <!-- String, max. 65 characters -->
        <sch:ValidFrom>2019-12-24T16:00:00</sch:ValidFrom>
        <sch:ValidUntil>2020-01-01T23:59:00</sch:ValidUntil>
            <!-- XSD DateTime data type format -->
            <!-- If a holiday lasts the whole day,
                enter 00:00 as the starting time and 23:59 as the end time. -->
        <sch:AppliesToCarriers>true</sch:AppliesToCarriers>
        <sch:EntranceIdList>
            <sch:EntranceId>1</sch:EntranceId>
                <!-- Long. You can find this ID with findEntrance -->
        </sch:EntranceIdList>
    </sch:HolidayAdd>
</soapenv:Body>
```

*Example code for **addHoliday**.*

```
<HolidayResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
    <Id>1</Id>
    <Name>Christmas break</Name>
    <ValidFrom>2019-12-24T16:00:00</ValidFrom>
    <ValidUntil>2020-01-01T23:59:00</ValidUntil>
    <AppliesToCarriers>true</AppliesToCarriers>
    <EntranceIdList>
        <EntranceId>1</EntranceId>
    </EntranceIdList>
</HolidayResult>
```

*Response for the previous code.*




*Result in AEOS.*

## 4.8 Automatically unlock public doors during office hours

With automatic schedules, you can automatically unlock or lock an entrance during a specific time of the day.

See the AEOS terminology overview (6.1) for an explanation of automatic lock/unlock schedules, verification disabled schedules and toggle schedules.

1. Find the entrance (4.3.2) that you want to apply a schedule to.
2. Find the Day/Time schedule (4.6.2) that you want to use for the automatic schedule.
3. Use the function **changeEntrance**.

   The **Id** and **UnitOfAuthType** fields are mandatory.

   The **scheduleid** fields are optional, only use the fields for the schedules that you want to use.

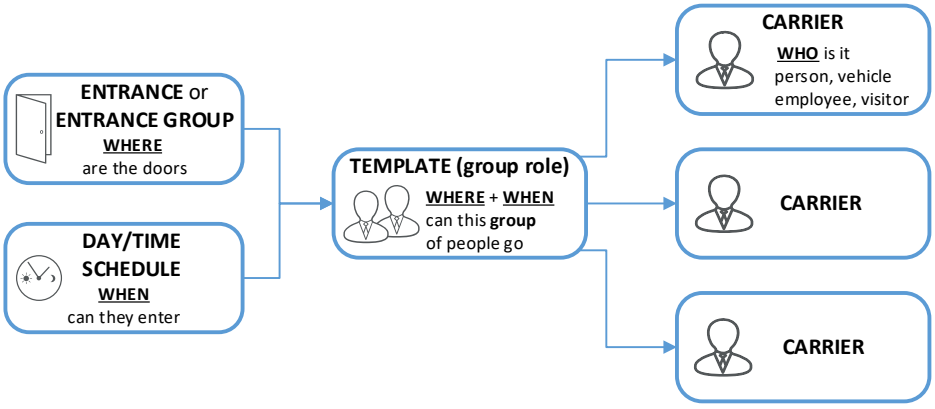
```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.nedap.com/aeosws/schema">
    <soapenv:Header/>
    <soapenv:Body>
        <sch:EntranceChange>
            <sch:Id>151</sch:Id>
            <sch:UnitOfAuthType>OnLine</sch:UnitOfAuthType>
                <!-- For the above 2, use the values from findEntrance -->
            <sch:verificationdisabledschedid>120</sch:verificationdisabledschedid>
            <sch:autounlockscheduleid>120</sch:autounlockscheduleid>
            <sch:autolockscheduleid>169</sch:autolockscheduleid>
            <sch:togglescheduleid>120</sch:togglescheduleid>
                <!-- For any of the above 4, use the value from findDayTimeSchedule -->
        </sch:EntranceChange>
    </soapenv:Body>
</soapenv:Envelope>
```

*Example code for **changeEntrance**. This code adds automatic schedules to an entrance.*

| Name ▲ | Location | Description | Automatic lock... | Automatic unlock... | Toggle schedule | Verification disable |
|---|---|---|---|---|---|---|
| Front entrance | Building 1 | Sliding doors | After hours | Working hours | Working hours | Working hours |

*Result in AEOS.*

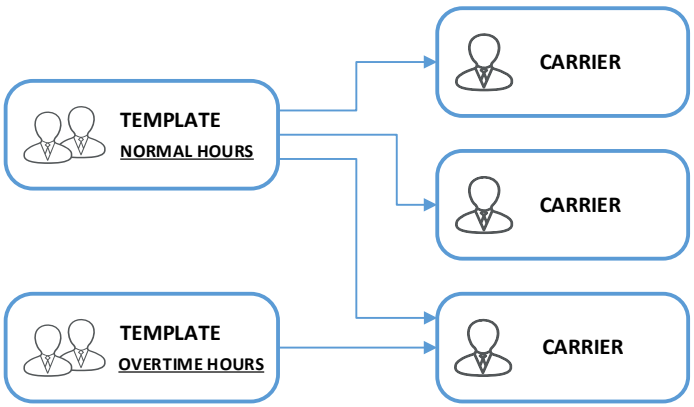
## 4.9 Templates




Templates define the authorisations in AEOS: where can the carriers go, and when. They consist of one or more **entrances** or entrance **groups**, combined with one **day/time schedule**.

**If your system runs in different time zones, AEOS uses the local time for day/time schedules.**

Templates are defined for groups of people, usually based on their role or department. This makes it easier to assign authorisations based on the function or area someone works in, so that these authorisations do not need to be assigned to every carrier individually.




You can assign several templates to a single carrier. For example, if you have a template for normal working hours and a template for overtime hours, you could assign both templates to employees who are allowed to work overtime, and only the normal working hours template to everyone else.

### 4.9.1 Create a template

1. Find the entrances (4.3.2), the entrance groups (4.5.2), and the day/time schedules (4.6.2) that you want to use for this template.
2. Use the function **addTemplate**.

   The **Name** and **UnitOfAuthType** fields are mandatory.

   The **ID** field will be generated by the system.

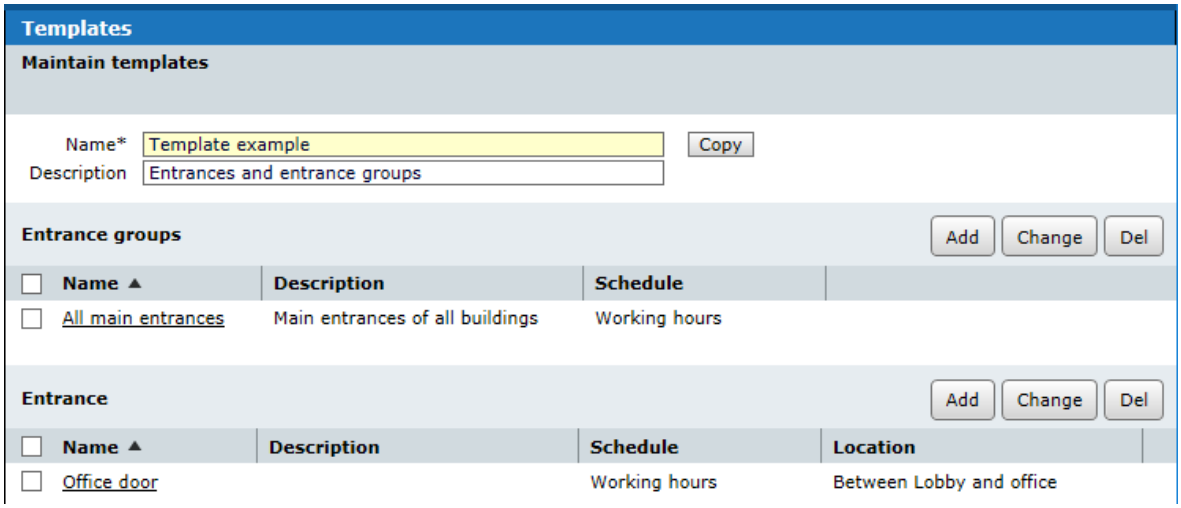
See the AEOS terminology overview (6.1) for additional explanations of templates, day/time schedules and online/offline doors.

```
<soapenv:Body>
    <sch:TemplateAdd>
        <sch:Name>Template example</sch:Name>
            <!-- String, max. 150 characters -->
        <sch:Description>Entrances and entrance groups</sch:Description>
            <!-- String, max. 150 characters -->
        <sch:UnitOfAuthType>OnLine</sch:UnitOfAuthType>
            <!-- Loxs, OffLine, OnLine -->
        <sch:TemplateItem>
            <sch:SubjectId>2</sch:SubjectId>
                <!-- Entrance ID, from findEntrance -->
            <sch:ScheduleId>120</sch:ScheduleId>
                <!-- day/time schedule ID, from findDayTimeSchedule -->
            <sch:AuthorisationType>Entrance</sch:AuthorisationType>
                <!-- Entrance, EntranceGroup, LoxsTerminalGroup -->
        </sch:TemplateItem>
        <sch:TemplateItem>
            <sch:SubjectId>101</sch:SubjectId>
                <!-- Entrance Group ID, from findEntranceGroup -->
            <sch:ScheduleId>120</sch:ScheduleId>
            <sch:AuthorisationType>EntranceGroup</sch:AuthorisationType>
        </sch:TemplateItem>
    </sch:TemplateAdd>
</soapenv:Body>
```

*Example code for **addTemplate**.*

```
<soap:Body>
    <TemplateResult xmlns:ns2="http://www.nedap.com/aeosws"
xmlns="http://www.nedap.com/aeosws/schema">
        <Id>102</Id>
        <Name>Template example</Name>
        <Description>Entrances and entrance groups</Description>
        <UnitOfAuthType>OnLine</UnitOfAuthType>
        <TemplateItem>
            <SubjectId>2</SubjectId>
            <ScheduleId>120</ScheduleId>
            <AuthorisationType>Entrance</AuthorisationType>
        </TemplateItem>
        <TemplateItem>
            <SubjectId>101</SubjectId>
            <ScheduleId>120</ScheduleId>
            <AuthorisationType>EntranceGroup</AuthorisationType>
        </TemplateItem>
    </TemplateResult>
</soap:Body>
```

*Response for the previous code.*




*Result in AEOS.*

## 4.9.2 Find a template

You can use this to find the ID of a template, which you need to change the template or add it to a carrier etc.

- Use the function **findTemplate**.
- The **UnitOfAuthType** field is mandatory, all other fields are optional.
- If you expect that this call returns a large number of items, you can specify the maximum amount of records to return (2.5.5).

```
<soapenv:Body>
    <sch:TemplateSearchInfo>
        <sch:TemplateInfo>
            <sch:UnitOfAuthType>OnLine</sch:UnitOfAuthType>
                <!-- OnLine, OffLine, Loxs -->
        </sch:TemplateInfo>
    </sch:TemplateSearchInfo>
</soapenv:Body>
```

*Example code for **findTemplate**. This code finds all online templates.*

## 4.9.3 Change a template

- Use the function **changeTemplate**.
- The **Id** and **UnitOfAuthType** fields are mandatory.
- All other fields are optional. Only specify the fields that you want to add or change.
  Any existing optional fields that are not specified in this call, are not deleted or changed.
- If you want to change or add a **TemplateItem,** specify the whole list, including any existing items. All existing **TemplateItem** entries are overwritten with the new list if a **TemplateItem** is included in the call.

See Create a template (4.9.1) for all available fields.

See the AEOS terminology overview (6.1) for additional explanations of templates, day/time schedules and online/offline doors.

# 5. Event descriptions

AEOS events are an AEOS internal feature, and can change their structure between AEOS versions without notice. There is no guarantee that code based on the below information will stay working in the future.

## 5.1 EventTypeId, EventTypeName and IntValue

| No. | Event type | IntValue | Description |
|---|---|---|---|
| 1 | AccessPointStateLockedEvent | Principle | The principle responsible for changing the access point state.<br>0 = unknown<br>1 = server command<br>2 = automatic lock schedule<br>3 = access point inputs |
| 2 | AccessPointStateNormalEvent | Principle | The principle responsible for changing the access point state.<br>0 = unknown<br>1 = server command<br>2 = automatic lock schedule<br>3 = access point inputs |
| 3 | AccessPointStateUnlockedEvent | Principle | The principle responsible for changing the access point state.<br>0 = unknown<br>1 = server command<br>2 = automatic lock schedule<br>3 = access point inputs |
| 4 | AuthorizationServiceIOEvent | - | Not used |
| 5 | BadgeAccessEvent | Direction | Indicates the direction of access.<br>0 = unknown<br>1 = in<br>2 = out |
| 7 | DirectDoorAlarmEvent | Begin/end | Indicates the begin or end of a direct door alarm.<br>0 = end<br>1 = begin |
| 8 | DoorOpenTooLongAlarmEvent | Begin/end | Indicates the begin or end of a door open too long alarm.<br>0 = end<br>1 = begin |
| 9 | LockSupervisorAlarmEvent | Alarm state | The lock supervisor alarm state.<br>0 = no alarm<br>1 = open<br>2 = shortcut |
| 10 | ManualUnlockedEvent | Direction and begin/end | Indicates the direction and begin/end status of a manual unlock. The rightmost 8 bits contain the begin/end status, the leftmost bits contain the direction.<br>Begin/end:<br>0 = end<br>1 = begin<br>Direction:<br>0 = unknown<br>1 = in<br>2 = out |
| 11 | NoAuthorizationServiceEvent | - | Not used |

| No. | Event type | IntValue | Description |
|---|---|---|---|
| 12 | InputContactEvent | Input state and number | Indicates the input state and input number. The rightmost 8 bits contain the input state, the leftmost bits contain the input number.<br>Input state:<br>0 = input passive<br>1 = input active<br>2 = input sabotaged open<br>3 = input sabotaged shortcut |
| 13 | BehaviourTestModeEvent | Test mode state | Indicates if the test mode is enabled or not.<br>0 = test mode disabled<br>1 = test mode enabled |
| 14 | DeviceConnectionEvent | Device connection state | Indicates if the device is currently connected.<br>0 = not connected<br>1 = connected |
| 15 | DeviceNetworkEvent | Device connection state | Indicates if the device is currently connected.<br>0 = not connected<br>1 = connected |
| 16 | AEPackDiscoveryEvent | - | Not used |
| 17 | AEPackRemovalEvent | - | Not used |
| 18 | AutomaticUnlockedEvent | Begin/end | Indicates the begin or end of an automatic unlock.<br>0 = end<br>1 = begin |
| 19 | DeviceIOEvent | Device connection state | Indicates if the device is currently connected.<br>0 = not connected<br>1 = connected |
| 20 | EmergencyUnlockedEvent | Begin/end | Indicates the begin or end of an emergency unlock.<br>0 = end<br>1 = begin |
| 21 | DoorOpenedInputSabotagedEvent | Input state | Indicates the input state value.<br>0 = input passive<br>1 = input active<br>2 = input sabotaged open<br>3 = input sabotaged shortcut |
| 22 | EmergencyUnlockInputSabotaged Event | Input state | Indicates the input state value.<br>0 = input passive<br>1 = input active<br>2 = input sabotaged open<br>3 = input sabotaged shortcut |
| 23 | ManualUnlockInputSabotaged Event | Input state | Indicates the input state value.<br>0 = input passive<br>1 = input active<br>2 = input sabotaged open<br>3 = input sabotaged shortcut |
| 24 | AEpuStatusEvent | Reachable state | Indicates if the aepu is reachable.<br>0 = not reachable<br>1 = reachable |

| No. | Event type | IntValue | Description |
|---|---|---|---|
| 25 | AccessPointModificationQuitEvent | Modification reason | 0 = access point set in locked mode<br>1 = access point set in normal mode<br>2 = access point set in unlocked mode<br>3 = access point added to the specified entrance<br>4 = access point removed from the specified<br>5 = entrance id for an access point set<br>6 = entrance id for an access point removed<br>7 = schedules for an access point set<br>8 = schedules for an access point removed<br>9 = entrance id and schedules for an access point<br>10 = access point set in emergency unlock mode<br>11 = access point set in emergency lock mode<br>12 = emergency mode undone |
| 26 | AccessPointModificationFailedEvent | Modification reason | 0 = access point set in locked mode<br>1 = access point set in normal mode<br>2 = access point set in unlocked mode<br>3 = access point added to the specified entrance<br>4 = access point removed from the specified<br>5 = entrance id for an access point set<br>6 = entrance id for an access point removed<br>7 = schedules for an access point set<br>8 = schedules for an access point removed<br>9 = entrance id and schedules for an access point<br>10 = access point set in emergency unlock mode<br>11 = access point set in emergency lock mode<br>12 = emergency mode undone |
| 27 | BooleanStateChangeEvent | Boolean state | Indicates the boolean state of this event.<br>0 = false<br>1 = true |
| 28 | LockInputSabotagedEvent | Input state | Indicates the input state value.<br>0 = input passive<br>1 = input active<br>2 = input sabotaged open<br>3 = input sabotaged shortcut |
| 29 | NormalInputSabotagedEvent | Input state | Indicates the input state value.<br>0 = input passive<br>1 = input active<br>2 = input sabotaged open<br>3 = input sabotaged shortcut |
| 30 | InhibitInputSabotagedEvent | Input state | Indicates the input state value.<br>0 = input passive<br>1 = input active<br>2 = input sabotaged open<br>3 = input sabotaged shortcut |
| 31 | InhibitEvent | Begin/end | Indicates the begin or end of an inhibit state.<br>0 = end<br>1 = begin |
| 32 | ApbGrantAccessEvent | Zone manager availability | Indicates if the zone manager was available.<br>0 = zone manager was not available<br>1 = zone manager was available |
| 33 | ApbCarrierResetEvent | Initiation principle | Indicates if the event is initiated by a system or by a user reset.<br>0 = initiated by user<br>1 = initiated by system |

| No. | Event type | IntValue | Description |
|---|---|---|---|
| 34 | NoBookingEvent | Direction | Indicates the direction.<br>0 = unknown<br>1 = in<br>2 = out |
| 35 | InputSabotagedEvent | Input state | Indicates the input state value.<br>0 = input passive<br>1 = input active<br>2 = input sabotaged open<br>3 = input sabotaged shortcut |
| 36 | CCFailureEvent | Error code | The error code belonging to this event.<br>0 = slide not opened<br>1 = slide open too long |
| 37 | CCillegalCardInsertedEvent | - | Not used |
| 38 | AntennaMonitorAlarmEvent | Alarm state | The alarm state of the antenna monitor.<br>0 = no alarm<br>1 = alarm |
| 39 | ThresholdGuardAlarmEvent | Alarm state | The alarm state of the threshold guard.<br>0 = below threshold<br>1 = at or above threshold |
| 40 | BadgeNoAccessEvent | Direction + reason | Indicates the direction and reason for no access. The rightmost 8 bits contain the direction value, the leftmost bits contain the reason.<br>Direction:<br>0 = unknown<br>1 = in<br>2 = out<br>Reason: see BadgeNoAccessEvent reasons (5.2). |
| 41 | VerificationAlarmEvent | Direction | Indicates the direction.<br>0 = unknown<br>1 = in<br>2 = out |
| 42 | InvalidVerificationEvent | Direction | Indicates the direction of access.<br>0 = unknown<br>1 = in<br>2 = out |
| 43 | InvalidVerifierEvent | Direction | Indicates the direction of access.<br>0 = unknown<br>1 = in<br>2 = out |
| 44 | NoAccessControlServiceEvent | - | Not used |
| 45 | ZoneChangedEvent | Begin/end | Indicates the begin or end of the zone state.<br>0 = end<br>1 = begin |
| 46 | StateChangedEvent | Begin/end | Indicates the begin or end of the state.<br>0 = end<br>1 = begin |
| 47 | IncompatibleAepuVersionEvent | - | Not used |
| 48 | ProvideAccessEvent | Direction | Indicates the direction.<br>0 = unknown<br>1 = in<br>2 = out |

| No. | Event type | IntValue | Description |
|---|---|---|---|
| 49 | PowerSupplyInputAlarmEvent | Alarm state | The alarm state of the power supply.<br>0 = power supply low<br>1 = power supply at or above threshold |
| 50 | PowerSupplyStateChangedEvent | Power supply state | Indicates the state of the power supply.<br>0 = init state<br>1 = mains + emergency<br>2 = mains + battery<br>3 = mains<br>4 = emergency<br>5 = battery<br>6 = unknown |
| 51 | CountGroupAlMostReached MaximumEvent | - | Not used |
| 52 | CountGroupMaximumNoLonger ReachedEvent | - | Not used |
| 53 | CountGroupMaximumReached Event | - | Not used |
| 54 | CountZoneAlMostReached MaximumEvent | - | Not used |
| 55 | CountZoneMaximumNoLonger ReachedEvent | - | Not used |
| 56 | CountZoneMaximumReached Event | - | Not used |
| 57 | AlarmSwitchedEvent | Switch action | Indicates if the alarm was switched on of off.<br>0 = switched off<br>1 = switched on |
| 58 | AlarmSwitchedForcedEvent | Switch action | Indicates if the alarm was switched on of off.<br>0 = switched off<br>1 = switched on |
| 59 | AlarmSwitchTimeOutEvent | Switch action | Indicates if the alarm was switched on of off.<br>0 = switched off<br>1 = switched on |
| 60 | AnalogMonitorAlarmEvent | Alarm state | The alarm state of the analog monitor.<br>0 = no alarm<br>1 = above maximum value<br>2 = below minimum value<br>3 = out of measuring range |
| 61 | CounterMinAlarmEvent | Value | The value of the counter |
| 62 | CounterMaxAlarmEvent | Value | The value of the counter |
| 63 | CountGrantAccessEvent | - | Not used |
| 64 | CounterChangedEvent | Value | The value of the counter |
| 65 | SIAEvent | - | Not used |
| 66 | DeviceDiscoveryEvent | - | Not used |
| 67 | DeviceRemovalEvent | - | Not used |
| 68 | AEpuApplicationStartedEvent | - | Not used |
| 69 | AEpuReloadedEvent | - | Not used |

| No. | Event type | IntValue | Description |
|---|---|---|---|
| 70 | ResetAllCountersEvent | - | Not used |
| 71 | NetMonitorAlarmEvent | Port number | Indicates the network port number for which the alarm was generated. |
| 72 | ResetCountZoneEvent | - | Not used |
| 73 | ArmStateEvent | Arm state | Indicates if the area is armed.<br>0 = not armed<br>1 = armed |
| 74 | ZoneInhibitedEvent | Start/end | Indicates the start or end of the zone inhibition.<br>0 = end<br>1 = start |
| 75 | ZoneIsolatedEvent | Start/end | Indicates the start or end of the zone isolation.<br>0 = end<br>1 = start |
| 78 | LoginEvent | Login/Logout | Indicates if a user logged in or logged out.<br>0 = logged out<br>1 = logged in |
| 79 | LoginFailedEvent | - | Not used |
| 80 | IncorrectVerifierEvent | - | Not used |
| 81 | InsufficientAccessLevelEvent | - | Not used |
| 82 | AEPackAltModeEvent | Alt mode | The alt mode state of an AEpack<br>0 = disabled<br>1 = enabled |
| 83 | PresenceTimeExceededEvent | - | Not used |
| 84 | MaxMovementsExceededEvent | - | Not used |
| 85 | ACConfigurationChangedEvent | - | Not used |
| 86 | VisitReleaseTimeExceededEvent | - | Not used |
| 87 | LockOccupationTimeOutAlarm Event | Stopped/started | Indicates if the occupation timeout stopped or started.<br>0 = stopped<br>1 = started |
| 88 | InhibitInputEvent | Input state and number | Indicates the input state and input number. The rightmost 8 bits contain the input state, the leftmost bits contain the input number.<br>Input state:<br>0 = input passive<br>1 = input active<br>2 = input sabotaged open<br>3 = input sabotaged shortcut |
| 89 | ZoneAlarmStateChangedEvent | Alarm state | Indicates the zone alarm state.<br>0 = unknown<br>1 = active<br>2 = passive |

| No. | Event type | IntValue | Description |
|---|---|---|---|
| 90 | UnlockedEvent | Direction and begin/end | Indicates the direction and begin/end status of the unlock. The rightmost 8 bits contain the begin/end status, the leftmost bits contain the direction.<br>Begin/end:<br>0 = end<br>1 = begin<br><br>Direction:<br>0 = unknown<br>1 = in<br>2 = out |
| 91 | DoorOpenedEvent | Begin/end | Indicates the begin or end of a door open event.<br>0 = end<br>1 = begin |
| 92 | SpeedMeasuredEvent | Speed | Indicates the measured speed. |
| 93 | BadgeRejectedByDeviceEvent | Reason | Indicates the rejection reason.<br>102 = carrier verified negative<br>103 = verification process was aborted<br>109 = carrier presented a fake verifier<br>110 = verification device reported a generic error<br>112 = verification type not supported<br>others = unknown reason |
| 94 | GuardTourStartedEvent | - | Not used |
| 95 | GuardTourStoppedEvent | - | Not used |
| 96 | GuardTourSuspendedEvent | - | Not used |
| 97 | GuardTourCompletedEvent | - | Not used |
| 98 | GuardTourTooFastEvent | - | Not used |
| 99 | GuardTourTooSlowEvent | - | Not used |
| 100 | GuardTourMissedDemarcation PointEvent | - | Not used |
| 101 | GuardTourResumedEvent | - | Not used |
| 102 | TotalGuardTourTooFastEvent | - | Not used |
| 103 | TotalGuardTourTooSlowEvent | - | Not used |
| 104 | AEPackMessageEvent | - | Not used |
| 105 | UserActionEvent | User action id | Indicates the user action id.<br>1 = login<br>2 = logout<br>others = command execution |
| 106 | FallBackModeEvent | Stopped/started | Indicates if the fallback mode was stopped or started<br>0 = stopped<br>1 = started |
| 107 | RmiLoginEvent | Login state | Indicates the login state.<br>1 = login<br>2 = logout<br>3 = timed out |
| 108 | ActionOnTokenAssignmentAlarm | - | Not used |
| 109 | ActionOnVerificationExclusion Alarm | - | Not used |

| No. | Event type | IntValue | Description |
|---|---|---|---|
| 110 | ActionOnApbExclusionAlarm | - | Not used |
| 111 | CarrierDateFieldExpirationAlarm | - | Not used |
| 112 | MaxThresholdExceededEvent | - | Not used |
| 113 | ActionOnProfileAlarm | - | Not used |
| 114 | ActionOnCarrierAlarm | - | Not used |
| 115 | SilentAlarm | - | Not used |
| 116 | ActionOnTemplateAlarm | - | Not used |
| 117 | ActionOnEntranceGroupAlarm | - | Not used |
| 118 | FallBackModeACDataLoadEvent | Action value | Indicates the action that belongs to this event.<br>0 = started<br>1 = completed<br>2 = canceled |
| 119 | LicenseExpiredEvent | Expiry type | Indicates the expiry type of the license.<br>0 = license expired on<br>1 = license renewal period<br>2 = license reset period |
| 120 | IMSConnectionEvent | Connection state | Indicates the connection status<br>0 = (re)connected<br>1 = connection lost |
| 121 | LockerDoorStateEvent | Locker number | The number of the locker. |
| 122 | LockerOccupiedEvent | Locker number | The number of the locker. |
| 123 | LockerPresenceEvent | Locker number | The number of the locker. |
| 124 | LockerTerminalPresenceEvent | Locker number | The number of the locker. |
| 125 | LockerSabotageAlarmEvent | Locker number | The number of the locker. |
| 126 | LockerOpenTooLongAlarmEvent | Locker number | The number of the locker. |
| 127 | AreaArmStateEvent | User id | The id of the user that performed the action. |
| 128 | AlarmStateEvent | - | Not used |
| 129 | BypassStateEvent | User id | The id of the user that performed the action. |
| 130 | TamperStateEvent | - | Not used |
| 131 | LockerBadgeEvent | Locker number | The number of the locker. |
| 132 | KeyAccessEvent | Taken/returned | Indicates if the key was taken or returned<br>0 = returned<br>1 = taken |
| 133 | ExternalCounterEvent | - | Not used |
| 134 | BadgeQueueActionEvent | - | Not used |
| 135 | KNXDatapointGetValue CommandEvent | - | Not used |
| 136 | KNXDatapointSetValue CommandEvent | - | Not used |
| 137 | OfflineBadgeAccessEvent | Direction | Indicates the direction.<br>0 = unknown<br>1 = in<br>2 = out |

| No. | Event type | IntValue | Description |
|---|---|---|---|
| 138 | OfflineBadgeNoAccessEvent | Direction | Indicates the direction.<br>0 = unknown<br>1 = in<br>2 = out |
| 139 | OfflineBatteryLowLevelEvent | - | Not used |
| 140 | LookupServerDiscoverEvent | - | Not used |
| 141 | StartStopTestLogbookEntry | - | Not used |
| 142 | InhibitLogbookEntry | - | Not used |
| 143 | IsolateLogbookEntry | - | Not used |
| 144 | AlarmRestoreLogbookEntry | - | Not used |
| 145 | AreaConfigLogbookEntry | - | Not used |
| 146 | ProfileConfigLogbookEntry | - | Not used |
| 147 | RouteConfigLogbookEntry | - | Not used |
| 148 | ScheduleConfigLogbookEntry | - | Not used |
| 149 | UserConfigLogbookEntry | - | Not used |
| 150 | AlarmLogbookEntry | - | Not used |
| 151 | SystemLoginLogbookEntry | - | Not used |
| 152 | SystemPasswordLogbookEntry | - | Not used |
| 153 | UILoginDisabledLogbookEntry | - | Not used |
| 154 | UserActionLogbookEntry | - | Not used |
| 155 | MandatoryUserActionLogbook Entry | - | Not used |
| 156 | OverrideLogbookEntry | - | Not used |
| 157 | ArmDisarmLogbookEntry | - | Not used |
| 158 | AutoInhibitLogbookEntry | - | Not used |
| 159 | RFLockDoorLeftOpenAlarmEvent | LockId | The id of the lock. |
| 160 | RFLockIntrusionAlarmEvent | LockId | The id of the lock. |
| 161 | GalaxyZoneStateEvent | - | Not used |
| 162 | GalaxyGroupStateEvent | - | Not used |
| 163 | GalaxyZoneAlarmStateEvent | Zone number | The number of the zone that generated the event. |
| 164 | GalaxyGroupAlarmStateEvent | Group number | The number of the group that generated the event. |
| 165 | FireSystemPanelEvent | Panel number | The number of the panel that generated the event. |
| 166 | FireSystemSensorEvent | Sensor number | The number of the sensor that generated the event. |
| 167 | FireSystemZoneEvent | Zone number | The number of the zone that generated the event. |
| 168 | FireSystemModuleEvent | Module number | The number of the module that generated the event. |
| 169 | SoaaCardUpdateSuccessfulEvent | - | Not used |

| No. | Event type | IntValue | Description |
|---|---|---|---|
| 170 | SoaaCardUpdateFailedEvent | Reason | Indicates the failed reason.<br>100 = identifier not assigned to carrier<br>600 = no SOAA authorizations for carrier<br>601 = badge removed from updater<br>602 = presented badge contains an unsupported SOAA version<br>603 = error while reading from card<br>604 = error while writing to card<br>605 = authorization data does not fit on card<br>606 = blacklist data does not fit on card<br>607 = identifier creation error<br>608 = SOAA authorization data is invalid<br>609 = presented card is not a SOAA card |
| 171 | SoaaLockBatteryLowEvent | LockId | The id of the lock. |
| 172 | SoaaLockJammedEvent | LockId | The id of the lock. |
| 173 | SoaaLockBatteryReplacedEvent | LockId | The id of the lock. |
| 174 | SoaaLockSystemEvent | LockId | The id of the lock. |
| 175 | SoaaLockInternalErrorEvent | LockId | The id of the lock. |
| 176 | SoaaLockFailedToUnlockEvent | LockId | The id of the lock. |
| 177 | SoaaLockTamperEvent | LockId | The id of the lock. |
| 178 | SoaaLockBlackListedCardDetectedEvent | LockId | The id of the lock. |
| 179 | SoaaLockBlacklistFullEvent | LockId | The id of the lock. |
| 180 | SoaaLockAccessGrantedEvent | LockId | The id of the lock. |
| 181 | SoaaLockAccessDeniedEvent | LockId | The id of the lock. |
| 182 | RegistrationEvent | Direction | Indicates the direction.<br>0 = unknown<br>1 = in<br>2 = out |
| 183 | AutoStopTestLogbookEntry | - | Not used. |
| 184 | SetSequenceAbortedLogBookEntry | - | Not used. |
| 185 | InstallerModeStartStopLogbookEntry | - | Not used. |
| 186 | PACandLogInputChangeLogbookEntry | - | Not used. |
| 187 | ContainerModificationEvent | - | Not used. |
| 188 | PasswordChangedEvent | - | Not used. |

## 5.2 BadgeNoAccessEvent reasons

### 5.2.1 Splitting the reason and direction values

In case of a **BadgeNoAccessEvent**, the **IntValue** contains both the direction and the reason. The rightmost 8 bits are used for the direction value and the other bits contain the reason. To retrieve the direction and reason from the intvalue, follow the procedure below:

1. Divide the intvalue by 256 to right shift the value by 8 bits.
2. In the resulting value, the number left of the decimal point is the reason.
3. The decimal part of the value is the direction value divided by 256.
4. Multiply the decimal part by 256 to retrieve the direction.

**Example:**

- IntValue is 26113.
- Dividing by 256 gives 102.00390625.
- Multiplying 0.00390625 by 256 gives 1.

The reason is 102 (carrier verified negative) and the direction is 1 (in).

### 5.2.2 Reason descriptions

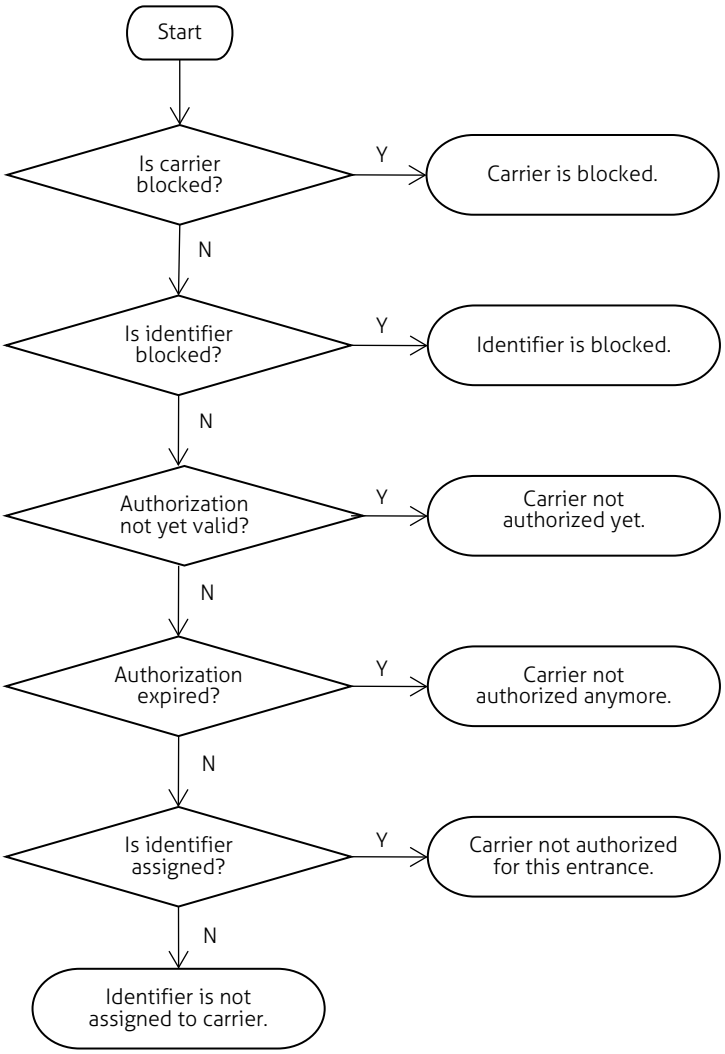
| Reason | Description |
|---|---|
| 0 | Verification has no result. |
| 3 | Verification alarm triggered. |
| 4 | Authorization has no result. |
| 100 | No authorization. The reason can not be determined by the AEpu, but is done at the server when this event is received. See Splitting the reason and direction values (5.2.1).<br>Underlying reason can be:<br>- unassigned badge<br>- no authorization for this entrance<br>- carrier is blocked<br>- identifier is blocked<br>- carrier is not authorized yet<br>- carrier is not authorized anymore |
| 101 | Identifier assignment not yet valid or expired |
| 102 | Carrier verified negative. |
| 103 | Verification process aborted. |
| 104 | Verification device error or not available. |
| 105 | Verification device does not know person. |
| 106 | No verification code for carrier available. |
| 107 | Blocked by security level. |
| 109 | Fake verifier presented. |
| 110 | Verification device reported a generic error. |

| Reason | Description |
|---|---|
| 112 | Unsupported verification type. |
| 197 | Internal error (authorizer). |
| 198 | Internal error (entrance). |
| 199 | Internal error (identifier). |
| 200 | No schedule available for this carrier and entrance. |
| 201 | Valid schedule, but schedule denies access. |
| 299 | Internal error (time schedule). |
| 300 | APB: invalid direction. |
| 301 | APB: entrance not known by zone manager. |
| 302 | APB: authorization request already running. |
| 303 | APB: carrier should not be present in this zone. |
| 304 | APB: zone manager unavailable. |
| 305 | APB: incorrectly configured AEpu. |
| 306 | APB: no access because of blocking time. |
| 400 | Counting: unknown carrier. |
| 401 | Counting: unknown entrance. |
| 402 | Counting: counter reached maximum value. |
| 403 | Counting: direction not specified. |
| 404 | Counting: unknown count group and zone combination. |
| 405 | Counting: count manager unavailable. |
| 500 | Lock occupied. |
| 501 | Lock time out alarm. |
| 505 | Locker is blocked. |
| 506 | Locker assignment is not allowed. |
| 550 | Authorization from external source is not possible due to missing user data attributes. |
| 551 | External authorization denied. |
| 552 | Authorization from external source is not possible due to a communication error. |
| 570 | Authorization from key cabinet is denied because user has keys he should return. |
| others | Unknown reason. |

### 5.2.3 BadgeNoAccessEvent Reason 100 Attributes

Reason number '100' is used when the AEpu determines that the presented identifier its carrier has no authorization for the entrance. The AEpu does not know why there is no authorization, because those details are only stored on the server. When the server receives a `BadgeNoAccessEvent` with reason '100', it will look for the information for the specified identifier and carrier to find the reason.

The detailed information is shown in the **Attribute** tag. This tag contains text entries of the form `key<=>value`, where multiple entries are separated by `|||`. The key usually consists of one or two characters, which again is dependent on the event type involved. You can use the following table and flow scheme to find the reason for a BadgeNoAccessEvent with reason '100'.




| Attributes involved |
| --- |
| Attributes contain P<=>1 (carrier blocked, user defined block reason in attribute D<=>...), or attributes contain P<=>2 (carrier blacklisted) |
| Attributes contain B<=>blockreason, where blockreason is the user defined reason for blocking the identifier. |
| Attributes contain A<=>1. |
| Attributes contain E<=>1. |
| CarrierId column contains a value. |

# 6. Error codes

| | | | |
|---|---|---|---|
| 5 | Find failed | 60 | Find employee failed |
| 9 | Configuration error, check AEOS logs | 61 | Add employee failed |
| 10 | Find department failed | 62 | Change employee failed |
| 11 | Add department failed | 63 | Remove employee failed |
| 12 | Change department failed | 65 | Find contractor failed |
| 13 | Remove department failed | 66 | Add contractor failed |
| 15 | Find unit failed | 67 | Change contractor failed |
| 16 | Add unit failed | 68 | Remove contractor failed |
| 17 | Change unit failed | 70 | Find visitor failed |
| 18 | Remove unit failed | 71 | Add visitor failed |
| 20 | Find countgroup failed | 72 | Change visitor failed |
| 21 | Add countgroup failed | 73 | Remove visitor failed |
| 22 | Change countgroup failed | 75 | Find car failed |
| 23 | Remove countgroup failed | 76 | Add car failed |
| 25 | Find entrance failed | 77 | Change car failed |
| 26 | Add entrance failed | 78 | Remove car failed |
| 27 | Change entrance failed | 80 | Find day/time schedule failed |
| 28 | Remove entrance failed | 81 | Add day/time schedule failed |
| 30 | Find entrancegroup failed | 82 | Change day/time schedule failed |
| 31 | Add entrancegroup failed | 83 | Remove day/time schedule failed |
| 32 | Change entrancegroup failed | 85 | Find identifiertype failed |
| 33 | Remove entrancegroup failed | 86 | Find carrier block reasons failed |
| 35 | Find carriergroup failed | 87 | Find identifier block reasons failed |
| 40 | Find country failed | 88 | Find identifier replacement reasons failed |
| 41 | Find site failed | 90 | Find visit failed |
| 42 | Find subsite failed | 91 | Add visit failed |
| 45 | Find region failed | 92 | Change visit failed |
| 46 | Find area failed | 93 | Remove visit failed |
| 47 | Find organization failed | 100 | Find carrier states failed |
| 50 | Find template failed | 101 | Find carrier profile failed |
| 51 | Add template failed | 102 | Find carrier verification(s) failed |
| 52 | Change template failed | 103 | Find carrier token(s) failed |
| 53 | Remove template failed | 104 | Find images (photo/thumbnail) of the carrier failed |
| 55 | Find person failed | 105 | Find carrier with given token failed |

| | | | |
|---|---|---|---|
| 106 | Find carrier presence failed | 177 | Change area failed |
| 107 | Find carrier attachment failed | 178 | Remove area failed |
| 108 | Find token failed | 179 | Add organization failed |
| 109 | Find carriers presence failed | 180 | Change organization failed |
| 110 | Change a state of the carrier failed | 181 | Remove organization failed |
| 111 | Change carrier profile failed | 190 | Find security level type failed |
| 112 | Change carrier verification failed | 191 | Add security level type failed |
| 113 | Assign token to carrier failed | 192 | Change security level type failed |
| 114 | Add images (photo) of the carrier failed | 193 | Remove security level type failed |
| 115 | Remove images (photo) of the carrier failed | 195 | Find security level template failed |
| 116 | Add carrier authorizations failed | 196 | Add security level template failed |
| 117 | Remove carrier authorizations failed | 197 | Change security level template failed |
| 118 | Add carrier attachment failed | 198 | Remove security level template failed |
| 119 | Change carrier attachment failed | 200 | Find security level scenario failed |
| 120 | Delete carrier attachment failed | 201 | Add security level scenario failed |
| 121 | To change the status of a token failed | 202 | Change security scenario type failed |
| 122 | Replace one token by another failed | 203 | Remove security scenario type failed |
| 123 | To block a token failed | 205 | Activate security scenarios failed |
| 124 | To unblock a token failed | 207 | Set a carrier group on carrier(s) failed |
| 125 | To withdraw a token failed | 210 | Find vendor failed |
| 126 | To withdraw and block a token failed | 211 | Add vendor failed |
| 130 | Find FreeField Definition failed | 212 | Change vendor failed |
| 131 | Find FreeField category failed | 213 | Remove vendor failed |
| 150 | Find eventtype failed | 215 | Find permit failed |
| 151 | Find events failed | 216 | Add permit failed |
| 160 | Find logtype failed | 217 | Change permit failed |
| 161 | Find logs failed | 218 | Remove permit failed |
| 170 | Add country failed | 219 | Withdraw permit from contr. failed |
| 171 | Change country failed | 220 | Find type-of-work failed |
| 172 | Remove country failed | 230 | Find holiday failed |
| 173 | Add region failed | 231 | Add holiday failed |
| 174 | Change region failed | 232 | Change holiday failed |
| 175 | Remove region failed | 233 | Remove holiday failed |
| 176 | Add area failed | 240 | Set token authenticationKey failed |

| | | | |
|---|---|---|---|
| 241 | Reset token authenticationKey failed | 273 | Set countzone amount failed |
| 242 | Add token with authenticationKey failed | | |
| 245 | Find access point failed | 280 | Find apbzone failed |
| 246 | Find unconfirmed access point failed | | |
| 247 | Confirming the access point(s) has failed | 285 | Find entrancezone failed |
| 248 | Change accesspoint failed | | |
| 250 | Find carrier countgroup assignments failed | 301 | Undefined UnitOfAuthorizationType |
| 251 | Add carrier countgroup assignments failed | 302 | Duplicate sort field |
| 252 | Remove carrier countgroup assignments failed | 303 | Invalid/Unknown Direction |
| 255 | Assign contactperson(s) failed | 310 | Find LoXS locker failed |
| 256 | Withdraw contactperson(s) failed | 315 | Find LoXS terminal failed |
| 260 | Assign sponsor(s) failed | | |
| 261 | Withdraw sponsor(s) failed | 901 | Not authorised |
| 265 | Find countzone failed | 902 | No license |
| 266 | Add countzone failed | 903 | Function not yet implemented |
| 267 | Change countzone failed | 904 | Wrong function is used |
| 268 | Remove countzone failed | | |
| 270 | Find countzonemanager failed | | |
| 271 | Change countzone configuration failed | 990 | System error |
| 272 | Find countzone amount failed | | |

# 7. Reference

## 7.1 AEOS terminology overview

For information on the AEOS authorisation model, see the *About AEOS* chapter in the ***AEOS user manual*** (version 3.4.0.x or higher). That manual also contains information on terminology and optional AEOS functions that are not listed here.

The *AEOS user manual* is available at the bottom of the *documentation overview* page on the Nedap web portal: https://portal.nedapsecurity.com/aeos-documentation-overview.

- People and cars that are authorized to access specific physical areas are called **carriers** in AEOS, because they *carry* an identifier (such as a badge) with them.
  **Person** carriers can be divided into **visitors**, **employees** and **contractors**.
  Contractors can work for **vendor** companies that must have a **permit** to do work for your company.
  If visit management is enabled in AEOS, you can add several **visits** to each visitor.
- Badges are called **identifiers** in AEOS (**tokens** in some SOAP functions).
  Identifiers have an **identifier type** that determines what kind of badge it is.
- You can optionally use **verification** as an additional check to see if the person who presents a valid identifier is the actual owner of this identifier. For example, a PIN code as verification in addition to a badge (identifier). You can only add a verifier in *addition* to an identifier; a carrier will never get access without valid identifier.
- You can give people authorisations to access specific areas at specific times with **templates** that consist of **entrances** (doors) or **entrance groups**, combined with **date/time schedules**.
- You can open public entrances for everyone during office hours with an **automatic unlock schedule**, or close high-security entrances for everyone (including people with valid badges) at night with an **automatic lock schedule**. You can disable PIN verification during office hours (so that a PIN is only necessary to enter outside office hours) with a **verification disabled schedule**. With a **toggle schedule**, the first valid badge unlocks the door for everyone, while the next valid badge closes it again.
- During a **holiday**, all selected entrances have their automatic unlock schedules cancelled. This means that they do not open automatically for everyone during office hours, but only open for valid badges. If you select **applies to carriers** for a holiday, only valid badges of carriers who have **access during holidays** are accepted.
- You can block people automatically after a period of inactivity with **auto block**. The **inactive period** is set in AEOS by the system administrator.
- Entrances can have three hierarchical **locations labels** (also called physical entrance labels) to define their exact location: **Country > Site > Sub site**.
- Entrance groups can have three types of independent (not hierarchical) **entrance group labels**: **region**, **area** or **organization**.
- **Entrances** on the AEOS server consist of one or more **access points** that are defined on the AEOS door controllers.
- **Access points** determine the types of doors and the hardware that the AEOS door controllers are connected to. The door controllers report their defined access points to the **AEOS server** (AEserver), so that AEOS can find them. You need to **confirm** a reported access point in AEOS before you can add it to an entrance.

- Entrances define the **authorisation** in AEOS, while access points define the **door hardware** in the door controllers. You can add more than one access point to the same entrance. However, once you have added an access point to an entrance, the access point is no longer available for any of the other entrances.
- **Zones** are areas in buildings with entrances in between. Zones can include multiple rooms. AEOS uses zones to register who is in which area at any given moment.

  The **APB (Anti Pass Back)** function uses **APB zones**.

  The **Counting** function uses **count zones**, where the **count zone manager** does not allow more than the maximum number of carriers that belong to a specific **count group** to enter.

  The **Maximum presence time** and **Maximum number of movements** functions use **entrance zones**.
- **Online** doors have a live connection with their door controller, either wired or wireless. The door controller determines if the door opens for someone or not. Any authorization changes that are made in AEOS, are transferred immediately to the door controller, so that the local authorizations are always up to date.

  Online door authorisations consist of **templates**, **entrances** and **entrance groups**.
- **OSS-SO** (OSS Standard Offline, **Soaa** in SOAP) doors use offline locks. An offline lock does not have a connection to a door controller. Instead, the authorization for an offline lock is written on the badge. Each offline lock has its own ID. A badge gives someone access to a number of doors with specific IDs, for a limited amount of time (the **OSS-SO update interval**). Each carrier must present their badge to an **OSS-SO updater** within the update interval to renew their authorizations on the badge.

  In AEOS, offline locks are presented as **OSS-SO entrances**, which can be grouped into **OSS-SO entrance groups**. OSS-SO entrances and entrance groups cannot be added to templates. Instead, they are assigned to carriers directly, combined with day/time schedules that have **OSS-SO restrictions** applied (for example, OSS-SO only allows 1-day or 7-day schedules).

  OSS-SO and Online doors do not mix: you cannot add OSS-SO entrances to an online entrance group or template, or add online doors to an OSS-SO entrance group.
- **Offline** doors are the same as OSS-SO doors, but they use a protocol specific to *Salto* and *Simons Voss* offline locks. Offline door authorisations consist of **Offline templates**, **Offline entrances** and **Offline entrance groups**. You cannot use Online templates, entrances and entrance groups for Offline door authorisations.
- When a carrier presents an identifier to a reader, AEOS generates several **events** (with different **event types**) that are stored in the **event log** and shown in the **event monitor**, unless the carrier status is set to **invisible**. If some carriers are set to **special**, you can track them separately on the event monitor with **event filters** that are set to only show special carriers.
- Carriers, identifiers, departments and count groups can be part of a **Unit**. Units can be added to a **filter** so that AEOS **system users** (the people issuing badges etc.) can only see the people that belong to their own unit. Use this when different companies share the same building, so that system users of one company cannot see or change the information of another company.
- **Security Level Management** uses **security level scenarios** of freely definable **security level types**. The scenarios contain **security level templates** which consist of **security level entrance groups** and **security level carrier groups**.

- **Locker Management** controls **LoXS** lockers as if they were entrances, which can be grouped in **locker groups**. Carriers can then be assigned a locker, so that their badge opens that locker. Carriers can also be assigned a **master key** of one or more locker groups, so that their badge can open all the lockers in that group.
- The AEOS **door controllers** (AEpus) use configurable software modules that are called **behavioural components** (AEbcs) to define the connected hardware and its behaviour. The combination of all AEbcs and their settings on a door controller is called a **door configuration**. You can edit door configurations on the door controllers with the **AEmon** software.
- Door controllers (AEpus) only have two connections for card readers. If you need more card readers on the same door configuration, you can connect up to 31 AEOS **door interfaces** (AEpacks) to a door controller via RS485. Door interfaces have additional hardware and reader connections that are added to the door configuration of the door controller that they are connected to.

## 7.2 Restart the AEOS lookup server and application server services

**In some cases it is not necessary to restart the server itself, and logging out from AEOS and then logging in again is enough. Try if that works before you try to restart the server as shown below.**

To restart the AEOS lookup server service or the AEOS application server service, do as follows.

1. In Windows, press the ⊞ **Win+R** key and enter **services.msc**. to open the **Services** window.
2. In the **Services** window, right-click on **AEOS Lookup Server** (or **AEOS_LOOKUP**) and select **Restart**.
3. In the **Services** window, right-click on **AEOS Application Server** (or **AEOS_APPL**) and select **Restart**.

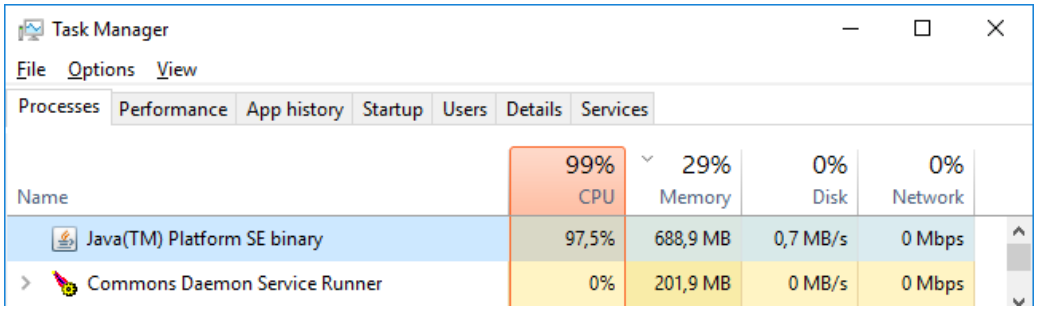
## 7.3 Find out if the Application Server has finished restarting

When you restart the AEOS Application Server or AEOS_APPL service, it takes some time before the AEOS web page can be found again. You can use the Task Manager to see if the server has finished starting up.

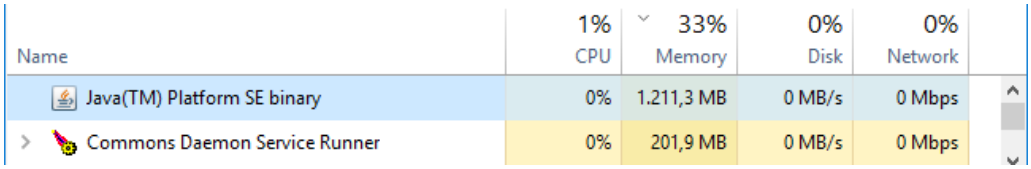
1. Right-click on the taskbar and select **Task Manager**.
2. Select the **Processes** tab.
3. In the list, look for the Java**(TM) Platform SE binary** item.

This item will have a high **CPU%**, and the used **Memory** will be increasing all the time.

At some point, the **Memory** will stop increasing (the value depends on the settings that were made during the installation of AEOS), and the **CPU%** value will become low. This means that the server has finished starting, and the AEOS web page is available again.




*Server is still starting:* Java(TM) *has a high* ***CPU%*** *and the* ***Memory*** *value increases*




*Server has finished starting:* Java(TM) *has a low* ***CPU%*** *and the* ***Memory*** *value stays the same*

## 7.4 Frequently asked SOAP questions

- **What SOAP functions are supported in AEOS?**
  This depends on the AEOS version. See WSDL function definitions.
- **Where can I find the WSDL file?**
  See WSDL function definitions.
- **What do all the terms used in the functions mean?**
  See AEOS terminology.
- **How do I install the WebService?**
  See Set up SOAP in AEOS.
- **When I add/change an entrance group/template an exception 'entrance not found' is returned.**
  Templates, entrance groups and entrances have either an OnLine or OffLine UnitOfAuthType.
  You cannot mix these types in a template or entrance group. For example, it is not allowed to put an online entrance in an offline entrance group or template.
- **Why does the example code from this manual not work when I copy it straight from this manual into SOAP?**
  For simplicity, some of the code examples do not include the envelope and header code, which is the same for every call. Make sure that this code is included in your call. See the add a person example (3.1.1) for a code example where the envelope and header are included.




**Nedap Security Management**
P.O. Box 103
NL - 7140 AC Groenlo
+31 (0)544 471 111
info@nedapsecurity.com
www.nedapsecurity.com