****

|  |
| --- |
|  |

**This project is funded by**

**the European Union**

|  |  |
| --- | --- |
| ECC Gateway - technical specification for declaranTs | |
| Version 1.2 | |
|  | |
|  | |
| **Author:** | Solitea a.s. |
| **Last changed:** | 28.4.2020 |
| **From:** | 11.2.2020 |
| **File name:** | 0-ECC GW Technical specification for declarants v1.2 |
| **Number of pages:** | 23 |
| **Confidentiality:** | Medium |

**Implementation of the New Computerized Transit System**



# PROJECT IDENTIFICATION

# Project data

|  |  |
| --- | --- |
| **Project Title** | Implementation of the New Computerized Transit System |
| **Contract Number** | CFCU/MNE/074 |
| **Contracting Authority** | Ministry of Finance, Directorate for Finance and Contracting of the EU Assistance Funds (CFCU) |
| **CA Programme Manager** | Ms. Vesna Lučić |
| **Beneficiary** | Montenegrin Customs Administration |
| **Beneficiary Contact person** | Mr Milan Martinović, Ms Tatjana Vujisić |
| **Contractor** | Consortium - Solitea Business Solutions s. r. o., SafeNet d. o. o., CoreIT d. o. o |
| **Contractor Contact person** | Mr Tomáš Vinš, Project Director |
| **Total budget of the action** | EUR 2,350,000.00 |
| **Total duration of the action** | 36 months |
| **Team Leader** | Miroslav Bujna |

**Status of document**

|  |  |  |
| --- | --- | --- |
| **Submitted on** | 11 February 2020 | |
| **Version** | 1.2 | |
| **Status** | Draft |  |
| For approval |  |
| Approved |  |

Document history

|  |  |  |
| --- | --- | --- |
| Version | Date | Description |
| 1.0 | 11. 2. 2020 | First version for MCA revisions |
| 1.1 | 22. 4. 2020 | Executive summary |
| 1.2 | 28. 4. 2020 | Executive summary and header update |

TabLE OF contents

[1. Executive summary 4](#_Toc38478126)

[2. ECC GW functions 5](#_Toc38478127)

[2.1. General overview of one business case processing 5](#_Toc38478128)

[2.2. Incoming message processing (SEND) 6](#_Toc38478129)

[2.3. Polling for new messages (POLL) 7](#_Toc38478130)

[2.4. Message delivering processing (DELIVER) 8](#_Toc38478131)

[2.5. Message confirmation processing (CONFIRM) 10](#_Toc38478132)

[3. Security requirements 11](#_Toc38478133)

[3.1.1. Authentication 11](#_Toc38478134)

[3.1.2. Authorization 11](#_Toc38478135)

[3.1.3. Integrity of data 11](#_Toc38478136)

[3.1.4. Confidentiality 11](#_Toc38478137)

[3.1.5. Digital signatures 11](#_Toc38478138)

[3.1.6. XML digital signature 11](#_Toc38478139)

[4. Structure of messages exchanged between external user and ECC GW 13](#_Toc38478140)

[4.1. ECC envelope 13](#_Toc38478141)

[4.2. Poll message 16](#_Toc38478142)

[4.3. ECCResponse message 17](#_Toc38478143)

[4.4. MessageIdentifier message 18](#_Toc38478144)

[5. List of error codes 19](#_Toc38478145)

[6. Example messages 22](#_Toc38478146)

[6.1. ECC Envelope 22](#_Toc38478147)

[6.2. Poll message 22](#_Toc38478148)

[6.3. ECCResponse message 23](#_Toc38478149)

[6.3.1. ACK 23](#_Toc38478150)

[6.3.2. NACK 23](#_Toc38478151)

[6.3.3. LIST 23](#_Toc38478152)

[6.3.4. ECC 23](#_Toc38478153)

[6.4. MessageIdentifier message 23](#_Toc38478154)

Table of figures

[Figure 1 - Logical schema of communication between external and national domain 4](#_Toc32330815)

[Figure 2 - general overview of one business case processing 6](#_Toc32330816)

[Figure 3 - Incoming message processing 7](#_Toc32330817)

[Figure 4 - Poll message processing 8](#_Toc32330818)

[Figure 5 - Message delivering 9](#_Toc32330819)

[Figure 6 - Message confirmation 10](#_Toc32330820)

1. Executive summary

This document is created as result of Activities related to Result 7: Electronic Customs Clearance Gateway of the EU funded project on “Implementation of the New Computerised Transit System (EuropeAid/139139/IH/SER/ME – Re-launch). The document is related to accepted ECC Gateway Functional and Technical Specification.

The document describes the functions of Electronic Customs Clearance Gateway (ECC GW) for external users. It is part of project delivery result 7, activity 7.4 defined by Inception report.

ECC GW provides secure interface for communication between Montenegrin Customs Administration (MCA) and external subjects (mainly economic operators, i.e. traders, declarants). It allows external subject to communicate with MCA in real time. The interface is implemented using various standard protocols and standards (HTTPS, SOAP, XMLDSIG, XAdES). Apart from transport functionality ECC GW also:

* verifies the integrity of transferred data (using digital signatures created by using of qualified certificates)
* performs validation against list of known certificates assigned to user which is sending the data
* communication between external domain (declarants) and ECC GW use https protocol, so all communication is encrypted at transport-level
* because business data itself lacks the information required to successfully process them in ECC GW, the data itself are encapsulated in another XML message (ECC envelope). ECC envelope will contain all the required parameters to successfully transfer the message



External Web Client

Figure 1 - Logical schema of communication between external and national domain

1. ECC GW functions

In all subsequent scenarios, where a communication on external ECC GW interface between ECC GW and external user is mentioned, it from technical point of view means communication through ECC GW proxy hosted on ESB platform, replicating 1:1 external ECC GW interface and ensuring additional network security measures.

All attributes, referred for authentication and authorization of external users and used by ECC GW (CommunicationAuthorizationID, Poll password, X509 certificates), are defined in related Communication Authorizations registered in AMS system (out of scope of this documentation).

* 1. General overview of one business case processing

The simplest business case consists of these steps:

* External user sends the ECC envelope with message (SEND functionality) which initiates business case (for example the message which lodges the declaration is sent) and receives ACK response
* External user periodically polls the ECC gateway in order to get the list of messages to be retrieved
* When ECC gateway sends non-empty list of messages waiting for delivery as response to POLL message, external user performs DELIVER and CONFIRM (in this order) to all of them, so the messages are delivered (DELIVER) and subsequently removed (CONFIRM) from queue of waiting messages

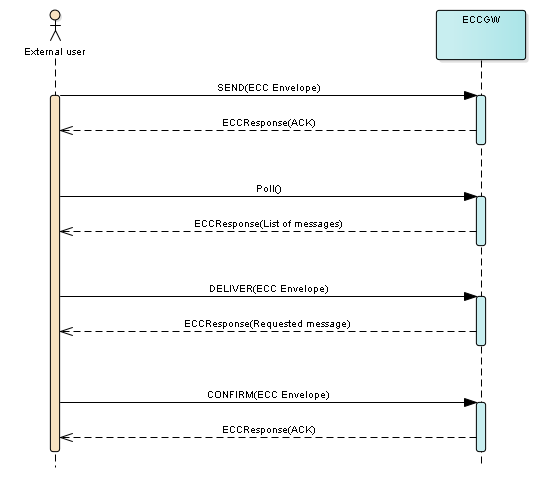


Figure 2 - general overview of one business case processing

* 1. Incoming message processing (SEND)

|  | **Incoming message processing** |
| --- | --- |
| **Description:** | ECC GW receives the ECC envelope from external. The ECC envelope encapsulates various business messages to be transferred to MCA |
| **Scenario(s):** | **Basic scenario (message is successfully received) – ECC GW component does these steps:**   1. Receives the ECC envelope 2. Verifies the envelope against its XSD (part of documentation) 3. Verifies digital signature of the message (using XMLDSIG and XAdES standards) 4. Verifies that certificate is valid – i.e. is not expired and was not revoked (using actual certificate revocation list (CRL)) 5. Extracts the encapsulated message and forwards it to corresponding national domain application. 6. Synchronously sends the positive acknowledgement message (ACK) back to calling system   **Alternative scenario (message is rejected) – ECC GW Core component does these steps:**   1. Receives the ECC envelope 2. Either XSD or digital signature or CRL validation fails 3. Stops further processing 4. Synchronously sends the negative acknowledgement message (NACK) back to calling system through security proxy placed on ESB platform |

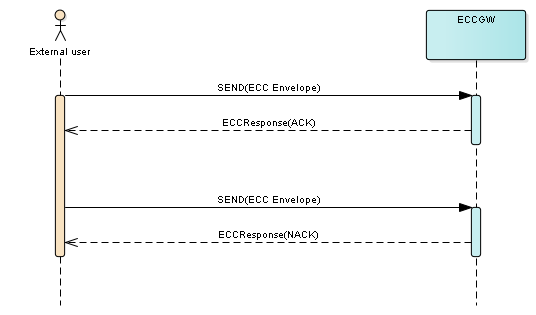


Figure 3 - Incoming message processing

* 1. Polling for new messages (POLL)

|  | **Poll message processing** |
| --- | --- |
| **Description:** | External user polls ECC GW in order to get the list of available messages which waits for further receiving |
| **Scenario(s):** | **Basic scenario (message is successfully processed):**   1. External user sends Poll message with the CommunicationAuthorizationID (unique ID which identifies external user) and Poll password (both CommunicationAuthorizationID and Password are defined in AMS system). Optionally also communication domain can be provided, in that case the response contains only messages regarding the specified communication domain. 2. ECC GW verifies that combination of CommunicationAuthorizationID and Poll password is correct 3. ECC GW responds with the list of messages which waits for receiving. If no such message exists, the returned list is empty   **Note:** Poll response contains only limited list of all waiting messages which can be downloaded by external user (sorted by date, oldest message first). This measure ensures that external user always performs deliver and confirm scenarios for all messages (without it the new messages do not appear in list).  **Alternative scenario (message is rejected):**   1. External user sends Poll message with the CommunicationAuthorizationID (unique ID which identifies external user) and Poll password 2. Either CommunicationAuthorizationID is unknown or the Poll password is not correct 3. ECC GW responds with NACK to the calling system |

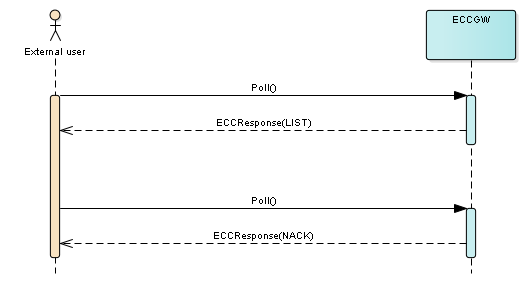


Figure 4 - Poll message processing

* 1. Message delivering processing (DELIVER)

|  | Message delivering processing |
| --- | --- |
| **Description:** | External user to receives the message from MCA |
| **Scenario(s):** | **Basic scenario (message is successfully processed):**   1. External user polls ECC GW using POLL function and receives non-empty list of messages 2. External user assembles the ECC for DELIVER function this way:    1. Header/@EnvelopeGuid is set to new unique identifier    2. Header/@Domain is set to domain of the message which is about to be delivered    3. Message/@MessageType is set to ADM001    4. Participans/Participant is set the very same way like in SEND scenario    5. Element Data contains message MessageIdentifier with element Identifier which contains Identifier from POLL response 3. External user sends the ECC assembled message using DELIVER function. 4. ECC GW receives ECC envelope and does the all needed checks. 5. ECC GW assembles the outgoing ECC envelope which contains the requested message and signs it. 6. ECC GW synchronously sends the assembled ECC envelope back to calling system.   **Alternative scenario (rejection):**   1. External user assembles the ECC which is not correct for some reason. 2. ECC GW receives the ECC envelope and performs the checks on it. 3. Either XSD or digital signature or CRL validation fails, or the message requested does not exist or is not meant for CommunicationAuthorizationID which was used in received ECC envelope, … 4. ECC GW assembles the ECCResult message with NACK and sends it synchronously back to calling system |

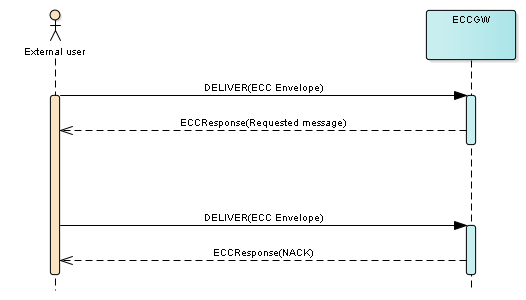


Figure 5 - Message delivering

* 1. Message confirmation processing (CONFIRM)

|  | Message confirmation processing |
| --- | --- |
| **Description:** | Provides the functionality for external user to confirm the successful delivery of the message from MCA. When message is confirmed, it is removed from ECC GW queue of messages to be delivered. |
| **Scenario(s):** | **Basic scenario (message is successfully processed):**  Steps are nearly the same like in DELIVER scenario, the only different things are:   * CONFIRM is used instead of DELIVER. * The response to CONFIRM is ACK, not the message to be delivered. * After successful CONFIRM the message is removed from outgoing messages queue, so it is no more offered in response to POLL message.   **Alternative scenario (rejection) – ECC GW does these steps:**   1. Receives the ECC envelope 2. Either XSD or digital signature or CRL validation fails, or the message requested is not existing or meant for CommunicationAuthorizationID which was used in received ECC envelope or message hasn’t been delivered yet 3. Assembles the ECCResponse message with NACK and synchronously sends it back to calling system |

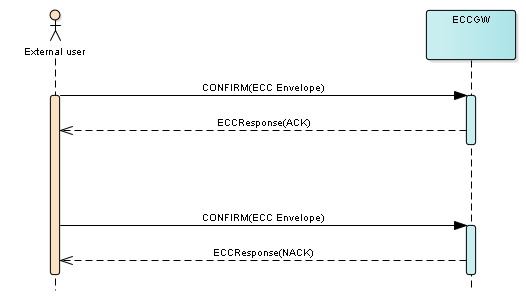


Figure 6 - Message confirmation

1. Security requirements
   * 1. Authentication

The identity of entity which digitally signed incoming message must be verified against list of allowed digital certificates belonging to CommunicationAuthorizationID specified in message. Alternatively by using CommunicationAuthorizationID with password (POLL message, chapter 2.3).

All information required for external user authentication is taken from MCA AMS (Authorization Management System).

* + 1. Authorization

Authorization of authenticated user ensures that user has the right to communicate with particular national domain system (like NCTS). All information required for external user authorization is taken from MCA AMS (Authorization Management System).

* + 1. Integrity of data

Integrity of data is achieved by using digital certificates and electronic signatures. This guarantees that data sent by external user are the same data which are received by ECC GW.

* + 1. Confidentiality

Confidentiality of transported data is guaranteed by using HTTPS for all communication between external user and ECC GW.

* + 1. Digital signatures

For authentication and authorization of external user, securing the data integrity and for non-repudiation of sent data digital signatures are used.

Electronic digital certificates will be used to electronically sign the ECC envelopes (which transfer business messages) in both directions. At trader’s side every person/system must use valid qualified electronic certificate issued by authorized qualified certification authority (QCA).

* + 1. XML digital signature

The specification of XML digital signatures is fully described by: <https://www.w3.org/TR/xmldsig-core/> . On top of this ECC GW requires XAdES-B-B or XAdES-B-T extension to be used. The XAdES extension is described here:   
<https://www.etsi.org/deliver/etsi_en/319100_319199/31913201/01.01.00_30/en_31913201v010100v.pdf>

Several facts/restrictions need to be taken into account:

* XML canonicalization of data to be signed has to be performed as a part of signing process (C14N without comments, preferably)
* XPath data transformation is forbidden (it’s not needed and creates high system load for complex messages)
* Strong SHA2 hash algorithms (SHA-256, SHA-384, SHA512) needs to be used as hashing algorithm used to compute digital signature
* Strong SHA2 hash algorithm needs to be used for asymmetric signature of data (for example <http://www.w3.org/2001/04/xmldsig-more#rsa-sha256>); usage of

<http://www.w3.org/2000/09/xmldsig#rsa-sha1> is forbidden

* Whole XML envelope (not just encapsulated message) is signed, so any manipulation made after the signature has been made will invalidate the signature
* From the view of W3C standard ECC GW uses “enveloped” form of signature, i.e. the Signature element is added as last child of root XML element
* ECC GW will be ready to produce outgoing messages (MCA -> external user) in XAdES-B-T format (which contains digital timestamp) by the time there will be suitable timestamping authority (TSA) to use by MCA

1. Structure of messages exchanged between external user and ECC GW
   1. ECC envelope

| **Element name** | | | | **XML type** | **Data type** | **Obligation for incoming messages** | **Obligation for outgoing messages** | **Description** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Header** | | | | Element |  | Mandatory[1] | Mandatory[1] | General envelope information |
|  | EnvelopeGuid | | | Attribute | UUID | Mandatory | Mandatory | Unique identifier of envelope. Each envelope must use unique value. |
|  | Domain | | | Attribute | String[20] | Mandatory | Mandatory | Communication domain (for example NCTS) |
| **Message** | | | | Element |  | Mandatory[1] | Mandatory[1] | Encapsulated message information |
|  | MessageType | | | Attribute | String[100] | Mandatory | Mandatory | Message type of encapsulated message (local-name of root element of encapsulated message (first XML element under Data element)) |
|  | PrimaryID | | | Attribute | String[100] | Optional | Optional | Primary business ID used in encapsulated message (for example for NCTS messages this elements is equal to MRN of declaration) |
|  | SecondaryID | | | Attribute | String[100] | Optional | Optional | Secondary business ID used in encapsulated message (for example for NCTS messages this elements is equal to LRN of declaration) |
|  | **Attributes** | | | Element |  | Optional | Optional | Optional generic message attributes; currently not used |
|  |  | **Attribute** | | Element |  | Mandatory[1-n] | Mandatory[1-n] | One attribute |
|  |  |  | Name | Attribute | String[20] | Mandatory | Mandatory | Attribute name |
|  |  |  | Value | Attribute | String[1000] | Mandatory | Mandatory | Attribute value |
| **Participants** | | | | Element |  | Mandatory[1] | Mandatory[1] | List of participants involved in message flow (currently only one – external user which sent the message) |
|  | **Participant** | | | Element |  | Mandatory[1-n] | Mandatory[1-n] | One of participants involved in message flow |
|  |  | CommunicationAuthorizationID | | Attribute | String[100] | Mandatory | Mandatory | Unique identification of external user. This value and the certificate used to sign the message are checked against AMS in order to authenticate external user and authorize further ECC envelope processing |
|  |  | ScenarioGuid | | Attribute | UUID | Mandatory | Mandatory | External user’s unique identification of one business case. Every message meant for the same business case should have the same ScenarioGuid, but messages for different business cases should have different ScenarioGuid. Used for correlation of business related messages |
|  |  | ApplicationName | | Attribute | String[20] | Mandatory | Mandatory | Identification (name) of external user application which sent the message. Together with application version (next row) can be used for example for finding the source of errors in message processing |
|  |  | ApplicationVersion | | Attribute | String[20] | Mandatory | Mandatory | Version of external user application which sent the message |
| **AdditionalInformation** | | | |  |  | Optional[1] | Optional[1] | Optional generic attributes which can be used in future to hold some additional information which cannot be sent in other ECC envelope elements/attributes |
|  | Attribute | | | Element |  | Mandatory[1-n] | Mandatory[1-n] | One attribute |
|  |  | Name | | Attribute | String[20] | Mandatory | Mandatory | Attribute name |
|  |  | Value | | Attribute | String[1000] | Mandatory | Mandatory | Attribute value |
| **Data** | | | | Element |  | Mandatory[1] | Mandatory[1] | This element encapsulates the message to be sent using ECC envelope (the root node of encapsulated message is the only child node of element Data) |
| **Signature** | | | | Element |  | Mandatory[1] | Mandatory[1] | Digital XML signature as described in chapter 3.1.6 |

* 1. Poll message

| **Element name** | **XML type** | **Data type** | **Obligation** | **Description** |
| --- | --- | --- | --- | --- |
| CommunicationAuthorizationId | Element | String[100] | Mandatory | Unique identification of external user |
| CommunicationDomain | Element | String[20] | Optional | Communication domain (for example NCTS). When given, only message to be delivered/confirmed appears in response |
| Password | Element | String[20] | Mandatory | Password for obtaining the list of messages (from AMS) |

* 1. ECCResponse message

| **Element name** | | | | **XML type** | **Data type** | **Obligation** | **Description** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ResponseType | | | | Element | String[10] | Mandatory | Type of response. One of the following values: ACK, NACK, LIST, ECC |
| **Messages** | | | | Element |  | Optional[1] | List of messages to deliver or confirm (used in response to Poll request (chapter 2.3) |
|  | Message | | | Element |  | Mandatory[1-99999] | One message to deliver or confirm |
|  |  | Identifier | | Attribute | UUID | Mandatory | Unique identifier of message which is about to be delivered and confirmed |
|  |  | Domain | | Attribute | String[20] | Optional | Communication domain of message (for example for NCTS) |
|  |  | MessageType | | Attribute | String[100] | Optional | Message type (usually name of message’s root element) |
|  |  | PrimaryID | | Attribute | String[100] | Optional | Primary business ID used in encapsulated message (for example for NCTS messages this elements is equal to MRN of declaration) |
|  |  | SecondaryID | | Attribute | String[100] | Optional | Secondary business ID used in encapsulated message (for example for NCTS messages this elements is equal to LRN of declaration) |
|  |  | Delivered | | Element | Bool | Mandatory | 0 or False = message has not been delivered yet; 1 or True = message was delivered and waits for confirmation of delivery from external user |
| **ECC** | | | | Element |  | Optional[1] | Root element of ECC envelope for message delivering (chapter 2.4) |
| **Errors** | | | | Element |  | Optional[1] | List of errors (used in NACK response) |
|  | **Error** | | | Element |  | Mandatory[1-n] | One error |
|  |  | | Code | Element | Integer | Mandatory | Numeric code of error |
|  |  | | Description | Element | String[1024] | Mandatory | Description of error |

* 1. MessageIdentifier message

| **Element name** | **XML type** | **Data type** | **Obligation** | **Description** |
| --- | --- | --- | --- | --- |
| Identifier | Element | UUID | Mandatory | Unique identifier of message which is about to be delivered or confirmed. To deliver/confirm the business message, external user’s system assembles this MessageIdentifier message, puts it into ECC envelope, signs it and then calls the appropriate functionality (deliver or confirm). Message/@MessageType attribute should be set to “ADM001”.  **Note:** This is ECC GW’s internal message which is processed entirely in ECC GW context. It’s not transferred to national domain applications (like NCTS). Since Identifier element is unique, it also means that this message is independent on communication domains used in ECC GW |

1. List of error codes

|  |  |  |  |
| --- | --- | --- | --- |
| Code | Operation(s) | Short description | Description |
| 0 | any | Unexpected error | Any error which is not covered by the codes bellow. |
| 101 | SEND | Envelope GUID is not unique | Every Header/@EnvelopeGuid value must be unique. |
| 102 | SEND, DELIVER, CONFIRM | Unknown domain | Domain specified in ECC envelope must be known one (for example NCTS). |
| 103 | SEND | Message type not equal to encapsulated message | Attribute Message/@MessageType must be equal to local-name of transferred message. |
| 104 | DELIVER, CONFIRM | Message is not in outgoing queue | Identifier used in MessageIdentifier message is unknown, or the message which is requested is not meant for the CommunicationAuthorizationID which was specified in DELIVER or CONFIRM message. |
| 201 | SEND, DELIVER, CONFIRM | Signature is not valid | Digital signature of XML message is not valid. |
| 202 | SEND, DELIVER, CONFIRM | Certificate is not valid | Certificate is not yet valid (= will be valid in future) or is expired. |
| 203 | SEND, DELIVER, CONFIRM | Certificate chain is not valid | One of the certificates in certificate chain is invalid. |
| 204 | SEND, DELIVER, CONFIRM | Certificate is revoked | Certificate used to sign the ECC envelope is found in CRL (Certificate Revocation List) |
| 205 | SEND, DELIVER, CONFIRM | General security error | There are multiple reasons for this error code:   * Message is not signed * Multiple “Signature” elements found in message * Invalid signature algorithm * Signature does not contain X509Certificate element with base64 encoded public key of signing certificate |
| 302 | SEND, DELIVER, CONFIRM | User is not authorized | Either CommunicationAuthorizationID is unknown (not specified in AMS) or the used certificate is not allowed for this CommunicationAuthorizationID. |
| 303 | SEND, DELIVER, CONFIRM | General authorization error | Other errors not covered by error 302. Reserved for future use. |
| 401 | POLL | User is not authorized | Either CommunicationAuthorizationID is unknown (not specified in AMS) or the used password is not equal to the password specified for this CommunicationAuthorizationID. |
| 501 | CONFIRM | Message was not delivered | Message cannot be confirmed without delivering it first. |

1. Example messages
   1. ECC Envelope

<ECC xmlns="http://www.carina.co.me/schemas/eccgw/v1">

<Header EnvelopeGuid="DB998947-6893-4EC9-B528-AA5413C09678" Domain="NCTS" />

<Message MessageType="ND015A" SecondaryID="LRN20200206193212703\_S002" />

<Participants>

<Participant CommunicationAuthorizationID="19ME123456790" ScenarioGuid="5B228E32-32C0-4A44-86C9-45FA85725CDA" ApplicationName="TestApp" ApplicationVersion="1.0" />

</Participants>

<Data>

<ND015A xmlns="">

…

</ND015A>

</Data>

<Signature Id="signature-18858181" xmlns="http://www.w3.org/2000/09/xmldsig#">

<SignedInfo>

<CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />

<SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />

<Reference Id="xadesReference-466024011" URI="">

<Transforms>

<Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />

<Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />

</Transforms>

<DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />

<DigestValue>wwLc/OqjJLHAXWD9tw7hyqcFfuM=</DigestValue>

</Reference>

<Reference URI="#xadesSignedProperties-601275841" Type="http://uri.etsi.org/01903#SignedProperties">

<DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />

<DigestValue>VucRJ3+P9BggYtlt6r4dxAq5rNE=</DigestValue>

</Reference>

</SignedInfo>

<SignatureValue>…</SignatureValue>

<KeyInfo>

<X509Data>

<X509Certificate>…</X509Certificate>

</X509Data>

</KeyInfo>

<Object>

<QualifyingProperties Target="#signature-18858181" xmlns="http://uri.etsi.org/01903/v1.3.2#">

<SignedProperties Id="xadesSignedProperties-601275841">

<SignedSignatureProperties>

<SigningTime>2020-02-06T18:32:12.8442462Z</SigningTime>

<SigningCertificate>

…

</SigningCertificate>

</SignedSignatureProperties>

<SignedDataObjectProperties>

<DataObjectFormat ObjectReference="#xadesReference-466024011">

<MimeType>text/xml</MimeType>

</DataObjectFormat>

</SignedDataObjectProperties>

</SignedProperties>

</QualifyingProperties>

</Object>

</Signature>

</ECC>

* 1. Poll message

<POLL xmlns="http://www.carina.co.me/schemas/eccgw/v1">

<CommunicationAuthorizationId>19ME123456EC00001</CommunicationAuthorizationId>

<CommunicationDomain>NCTS</CommunicationDomain>

<Password>pass123456</Password>

</POLL>

* 1. ECCResponse message
     1. ACK

<ECCResponse xmlns="http://www.carina.co.me/schemas/eccgw/v1">

<ResponseType>ACK</ResponseType>

</ECCResponse>

* + 1. NACK

<ECCResponse xmlns="http://www.carina.co.me/schemas/eccgw/v1">

<ResponseType>NACK</ResponseType>

<Errors>

<Error>

<Code>0</Code>

<Description>Unknown error</Description>

</Error>

</Errors>

</ECCResponse>

* + 1. LIST

<ECCResponse xmlns="http://www.carina.co.me/schemas/eccgw/v1">

<ResponseType>LIST</ResponseType>

<Messages>

<Message Identifier=”127cea8e-5d34-41b7-9bbb-4bc75212c4c6” Domain=”NCTS” MessageType=”ME015A”

SecondaryID=”ID12345” Delivered=”0”/>

</Messages>

</ECCResponse>

* + 1. ECC

<ECCResponse xmlns="http://www.carina.co.me/schemas/eccgw/v1">

<ResponseType>ECC</ResponseType>

<ECC>

<Header EnvelopeGuid="DB998947-6893-4EC9-B528-AA5413C09678" Domain="NCTS" />

<Message MessageType="ND015A" SecondaryID="LRN20200206193212703\_S002" />

<Participants>

<Participant CommunicationAuthorizationID="19ME123456790"

ScenarioGuid="5B228E32-32C0-4A44-86C9-45FA85725CDA" ApplicationName="TestApp"

ApplicationVersion="1.0" />

</Participants>

<Data>

<ND015A xmlns="">

<Header>

<LRN>LRN20200206193212703\_S002</LRN>

<FallbackProcedure>0</FallbackProcedure>

</Header>

…

</ND015A>

</Data>

<Signature Id="signature-18858181" xmlns="http://www.w3.org/2000/09/xmldsig#">

…

</Signature>

</ECC>

</ECCResponse>

* 1. MessageIdentifier message

<MessageIdentifier>

<Identifier>127cea8e-5d34-41b7-9bbb-4bc75212c4c6</Identifier>

</MessageIdentifier>