CEN/TC XBRL

Date: 2013-12

TC XBRL WI XBRL002

CEN/TC XBRL

Secretariat: NEN

Improving transparency in financial and business reporting — Metadata container

Verbesserung der Transparenz im finanziellen und geschäftlichen Reporting — Datencontainer mit Metadaten

Amélioration de la transparence du reporting financier et commercial — Conteneur de données avec données méta

ICS:

Descriptors:

Document type: CWA
Document subtype:
Document stage: Formal Vote
Document language: E

C:\EBA\CEN\FinalDocs\CWA-2\CWA_XBRL_WI002 (E).docx STD Version 2.5a

Contents

Foreword

This document has been prepared by CEN/WS XBRL, the secretariat of which is held by NEN.

This CWA is one of a series of related deliverables. The other deliverables are:

CWA XBRL 001 consists of the following parts, under the general title *Improving transparency in financial and business reporting — Harmonisation topics*:

— *Part 1: European data point methodology for supervisory reporting.*

— *Part 2: Guidelines for data point modelling*

— *Part 3: European XBRL Taxonomy Architecture*

— *Part 4: European Filing Rules*

— *Part 5: Mapping between DPM and MDM*

CWA XBRL 003-1 *Improving transparency in financial and business reporting — Standard regulatory roll-out package for better adoption — Part 1: XBRL Supervisory Roll-out Guide*

CWA XBRL 003-2 *Improving transparency in financial and business reporting — Standard regulatory roll-out package for better adoption — Part 2: XBRL Handbook for Declarers*

Introduction

The present document specifies a standard security envelope and an approach to integrate metadata usable for the European supervision authorities in order to receive reporting data in a standardised way. This standard has been elaborated over the years 2012 and 2013 and has been reviewed in a public consultation in the third quarter of 2013.

1 Scope

The purpose of this CWA is to propose a standard for submitting data instances to financial regulators in accordance with the chapter describing this CWA in the business plan [26]:

"*"Metadata container" to wrap a submitted XBRL instance document and compliance test. Provide a standard Metadata Container to enable XBRL sourcing, with in addition necessary compliance tools to enable all stakeholders to test and ensure full adherence to the technical standards.*

Metadata such as sender of the document, contact details, date and time of submission, version, digital signature, etc.. are not included in the taxonomies, because they really don't belong to the data model. On the other hand, and often for legal reasons, these data are required by national regulators. As a consequence, a variety of national protocols has been engineered, which complicates the life of cross-border institutions, but also prohibit the possibility to create a harmonized European collection system. Metadata are needed as well for financial reporting as for company legal and economical data. For the digital signature, existing solutions from the Business Registers, who have a deep expertise of the topic, may be generalized. In order to ensure compliance with the protocol, this project will deliver online tools for all stakeholders to use and to test compliance with the complete set (metadata container and XBRL instance document.

This CWA will provide standard protocols and mechanisms for digital signature, administrative data such as identification of submitter, feedback parameters, versioning of subsequent submissions and encryption, as well as online collaborative tools to ensure compliance."

This document specifies:

— a **submission container** structure to enable financial institutions to submit their regulatory reporting to the respective regulators in a standardised way;

— a **metadata** information structure (called « **Header** ») that is part of the submission container structure;

— an adequate negative (or positive) acknowledgement to be returned by the regulator to indicate if the submission container was well received by the regulator (or not);

— a **response container** structure to allow the regulator to return content-related error messages for the data instances in case errors occurred during any validation phase.

The main targeted authorities are the EBA (European Banking Authority) and EIOPA (European Insurance and Occupational Pensions Authority) as well as their related national supervision agencies, but the standard may also be used by other regulators. All container structures defined allow the packaging and securisation of data in a uniform way, which should lead to a greater transparency and interoperability between the declaring entities and the national and the European supervisory authorities.

In the course of the specification process, supplementary requirements were added by stakeholders or authorities concerned, among which:

— The scope of the data instances to be supported has been extended from pure XBRL instances to any type of structured data instances, including XML, CSV, etc.;

— The possibility of a 2-layer (or even multi-layer) submission process: some data instances are to be processed by the receiving authority itself (e.g. a national authority), others may be forwarded to a subsequent authority (e.g. a European one);

— The possibility of using the structures of the present CWA in a secure environment i.e. an environment that has its own signature and/or encryption facilities;

— The possibility of adding non-standard metadata if required (extensibility of the metadata header).

An important development approach for this CWA is to be flexible enough to support many different uses in different environments. For this reason some aspects (e.g. types of identifiers for financial institutions) could not be fixed by this standard and they shall be determined for every specific use of this standard via complementary instructions.

The present specification only defines the structures for the container itself, it does not define any transport aspects; the submission of a container may thus be freely combined with any type of transport protocol (submission via e-mail, (s)ftp, web portal, web services, …) in accordance with the local requirements.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ETSI Technical Report 102 038 v1.1.1, *Electronic Signatures and Infrastructures; XML format for signature policies*

ETSI Technical Specification 101 903 v1.4.1, *XML Advanced Electronic Signatures (XAdES)*

ETSI Technical Specification 102 176-1 v2.1.1, *Electronic Signatures and Infrastructures; Algorithms and Parameters for Secure Electronic Signatures; Part 1 Hash functions and asymmetric algorithms*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1
reporting entity
entity submitted to financial reporting and legally responsible for it

Note 1 to entry: (in many cases it uses internal resources to play the role of Content Producer and Technical Sender too). Also known as 'Declarer', 'Sender', '<ReportingEntity>'.

Note 2 to entry: An authority may also play the role of a reporting entity, e.g. when a national authority is providing data to a subsequent European authority as level-2 reporting.

3.2
technical sender
(potential) sub-contractor in charge of physically sending the data in respect of the present CWA (aware of containers, encryption, etc.)

Note 1 to entry: Also known as '<TechnicalSender>'.

3.3
content producer
(potential) sub-contractor in charge of the production of the content of the reporting and responsible for the accuracy of the content

Note 1 to entry: Also known as '<ContentProducer>'.

3.4
receiver
entity receiving reported data; also known as 'Authority' or 'Regulator' or 'Supervisor’

3.5
security envelope
XML structures surrounding the .zip file(s) after encryption and / or signature phase in accordance with the present CWA

3.6
negative acknowledge
information to the sender that the submission container could not be accepted because of error conditions (usually an instance of the « ContainerFeedback » schema with the tag <ContainerValidationFlag> having the value false

3.7
positive acknowledgement
information to the sender that the submission container has been accepted for processing of the data instances (usually an instance of the « ContainerFeedback » schema with the tag <ContainerValidationFlag> having the value true

3.8
instructions
supplementary information drafted by the receiver on how exactly to use the present CWA for a determined use.

3.9
certificate
a standard IETF X.509 digital certificate

Note 1 to entry: Public key should be RSA (rsaEncryption) with key length at least 2048.

3.10
header file
header file that complies with the « Header » schema

Note 1 to entry: See chapter 4.3.1.

3.11
container feedback file
container feedback file that complies with the « ContainerFeedback » schema

Note 1 to entry: See chapter 4.3.2.

3.12
instance feedback file
An instance feedback file that complies with the « InstanceFeedback » schema

Note 1 to entry: See chapter 4.3.3.

3.13
alternative instance feedback file
instance feedback file that is in another format than that of an instance of the « InstanceFeedback » schema

4 Files in containers

4.1 Introduction

The present chapter describes the files intervening in this standard, starting with simple files and continuing with the composed ones.

4.2 Data files

Data files are files that contain data, whether these data are structured of not. Data files can be any structured files like XBRL or XML instances, but also unstructured files like spread sheets or word processor files. The container controls files described in the next chapter as well as composed files (files that contain other files) are not part of the data files.

4.3 Container control files

The three types of container control files developed within this CWA are described in the following chapters.

4.3.1 Header file

A header file is an XML instance of an XML schema built according to the indications of chapter 6.3.3.

The function of the header file is to describe the global characteristics of the data files in the submission.

4.3.2 Container feedback files

A container feedback file is an XML instance of the XML schema located at:

http://www.eurofiling.info/eu/fr/esrs/ContainerFeedback/ContainerFeedback.xsd.

The function of the container feedback file is to confirm to the sender the success (or not) of the submission.

4.3.3 Instance feedback files

Instance feedback files are XML instances of the XML schema located at:

http://www.eurofiling.info/eu/fr/esrs/InstanceFeedback/InstanceFeedback.xsd.

Alternative representations of the error conditions of the data files submitted (e.g. documents with links to external systems representing the errors graphically, spread sheets with "red" cells indicating error locations, …) may be added to a response container, either as a complement or as an alternative to the XML instance feedback file. In that case the term alternative instance feedback file will be used.

4.4 ZIP compressed file

A Zip compressed file is a set of one or more files compressed together (ZIP [18]).

4.5 Secured files

The following chapters describe the files to which security operations have been applied.

4.5.1 Encrypted file

An encrypted file is a file embedded and encrypted in an XML instance of the XML schema (XMLENCR-CORE [14]).

4.5.2 Signed file

A signed file is a file embedded and signed in an XML instance of the XML schema (ETSI-XAdES [2]).

4.6 File naming conventions

The present CWA has defined the minimum required file naming conventions as described in the present chapter. The aim was to give the regulators vast degrees of freedom to define for their own purposes a file naming convention that serves best their requirements. So excepted for the reserved names and suffixes described in this chapter, the receiver's instructions may define adequate file naming conventions for containers, folders, data files etc.

4.6.1 Reserved file names

4.6.1.1 header.xml

The name « header.xml » is exclusively reserved for files of the type « header file ».

4.6.2 Instance feedback file name

The name and location of instance feedback files and other types of alternative instance feedback files should be chosen in such a way that the reconciliation of the feedback file with the corresponding data instance in the submission container is evident.

4.6.3 Reserved file name suffixes

All files shall have the « usual » file extension applicable in environments without restriction to the length of the extension: « .xbrl » for XBRL instances, « .xml » for XML instances, « .csv » for comma separated files etc.

4.6.4 Reserved extended suffixes

4.6.4.1 .signed.xml

The file extension « .signed.xml » is exclusively reserved for signed files.

4.6.4.2 .encrypted.xml

The file extension « .encrypted.xml » is exclusively reserved for encrypted files.

4.6.4.3 .containerfeedback.xml

The file extension « .containerfeedback.xml » is exclusively reserved for container feedback files complying with the ContainerFeedback schema.

4.6.4.4 .instancefeedback.xml

The file extension « .instancefeedback.xml » is exclusively reserved for instance feedback files complying with the InstanceFeedback schema.

5 Container

A container is a ZIP compressed file that contains a set of data files to be submitted.

A container may contain any type of files (e.g. other containers).

Folders may optionally be used in a container to better structure the files.

Folder conventions are not defined in this document.

5.1 Submission container

A submission container is a container that contains 1 header and 0 or more files and that is to transfer reporting data from the sender to the receiver.

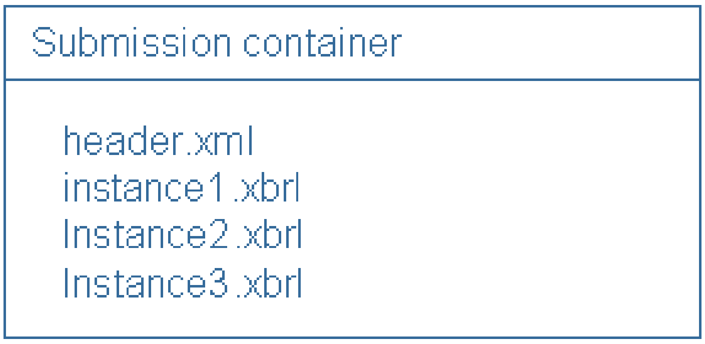


Figure 1 — Submission container example 1: Structure of a simple submission container with only one type of reporting in XBRL format and no use of folders

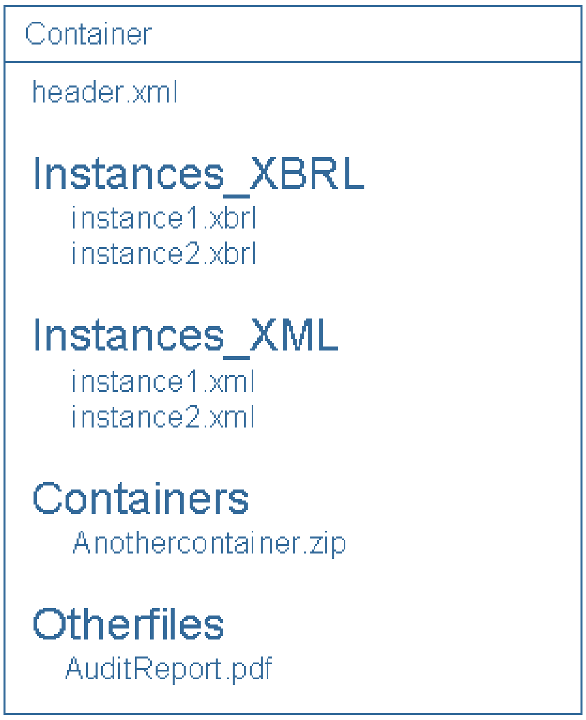


Figure 2 — Submission container example 2: Advanced structure of a submission container using folders (bold) to structure multiple types of reporting, containers, supplementary information etc.

5.2 Response container

A response container is a container that may be returned by the receiver of a submission container to its sender to inform the sender about the result of the evaluation of its content (e.g. possible errors).

When applicable (i.e. XBRL instance documents), XML Instances of the InstanceFeedback schema may be used to report the errors that were identified during the validation phase by the receiver, knowing that:

1) alternative instance feedback files are allowed as a replacement or as a supplement to the instance feedback files;

2) instance feedback files should be generated systematically, even if no errors at validation time occurred (not only negative, but also positive feedback should be provided for the data instances in the related submission container).

A response container is composed of the following files:

— 0 or 1 container feedback file;

— 0 to n instance feedback files and / or 0 to n alternative error feedback files.

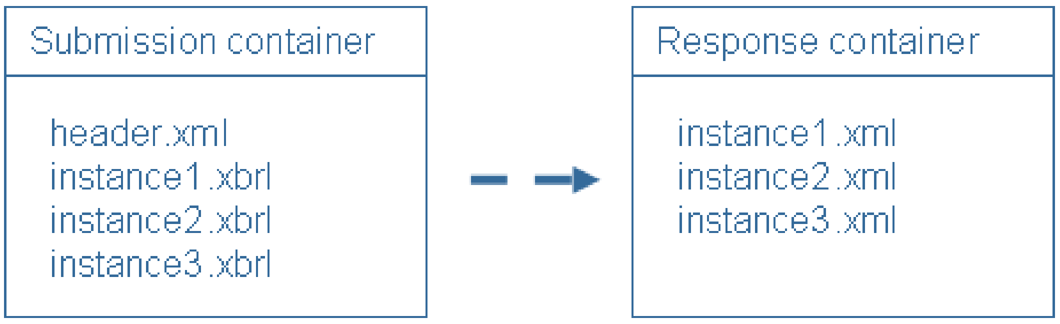


Figure 3 — Example of a response container generated on the basis of an incoming submission container with one reporting consisting of three XBRL files. All files in the response container are instances of the XML schema InstanceFeedback

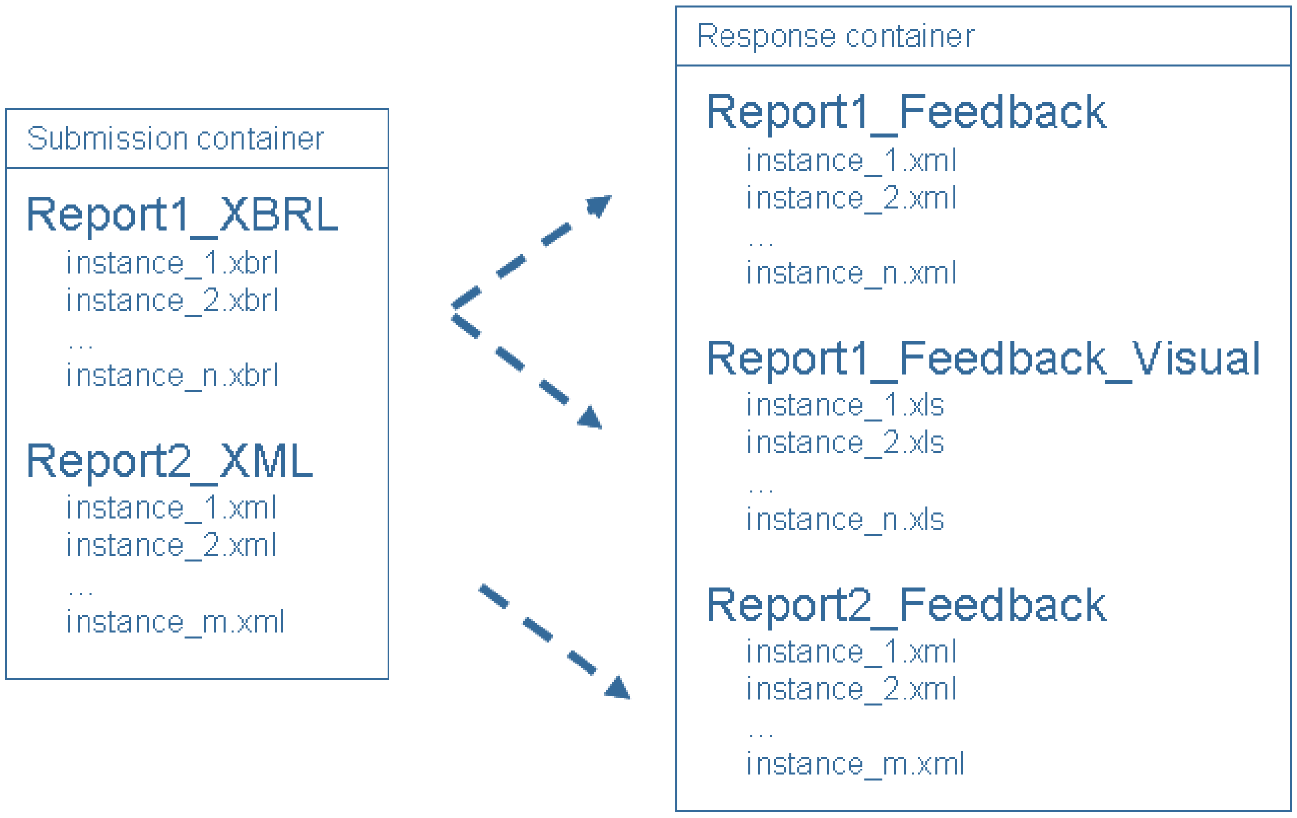


Figure 4 — Example of a response container generated on the basis of an incoming submission container with two different reportings and using folders. All XML files in the response container are instances of the XML schema InstanceFeedback. As a supplement, Excel-type error-diagnostics are returned for Report1

6 Primitive functions

The present chapter describes the primitive functions required to put in place the present CWA.

6.1 Compression functions

Compression is made in accordance with the ZIP file format specification [18].

The minimum feature version is 2.0 as defined in chapter 4.4.3.2 of the present version of the specification (version 6.3.3).

6.1.1 Creating a ZIP compressed file

Many tools in the market are able to create ZIP compressed files; interoperability problems are not known as long as multi-volume zip is not used. This is why multi-volume ZIP compressed files are not supported by this CWA version.

In order to avoid problems with senders using features of very recent versions not yet supported by the receiver, the instructions of the receiver may fix further constraints on the compression to use (e.g. a maximum level of the zip standard, as supported by the receiver).

6.1.2 Expanding a ZIP compressed file

This operation is the inverse operation of “Creating a ZIP compressed file”.

6.2 Security functions

This chapter describes the primitive functions for signing or encrypting files as well as the way to calculate the hash required in schemata InstanceFeedback and ContainerFeedback.

Within this specification, encryption and / or digital signature shall be applied to a single file (not to a set of files).

6.2.1 Encrypting a file

As references XMLENCR-CORE [14];

using key transport RSA-OAEP:

    http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p

and encrypting data with AES256:

    http://www.w3.org/2009/xmlenc11#aes256-gcm

The selected encryption uses the W3C XML Encryption to cipher a file, embedding it completely into the XML document that will result of the encryption process (there shall be no references to the file at an external location). Inside the CipherData element, there shall be a CipherValue element, but there shall not be a CipherReference element.

Basic steps for encryption are:

— create XML document with the embedded file, using W3C Encryption schema;

— generate AES-256 key (secret key);

— get RSA public key (and certificate);

— cipher secret key with public key, using RSA-OAEP;

— cipher XML element with the embedded file with AES-256 using secret key;

— store all in a file using W3C Encryption schema [14].

The embedded file is encrypted using a symmetric algorithm (AES-256) with a generated secret key. The security strength of AES-256 is 256 (NIST SP 800-57 part1 [19]).

The key transport algorithm RSA-OAEP with mask generation function MGF1 (MGF1p, padding) is used to cipher the generated AES-256 secret key. Key transport algorithms are public key encryption algorithms especially specified for encrypting and decrypting keys. RSA-OAEP uses the receiver’s public key to encrypt the secret key generated by AES while encrypting the file. This key transport algorithm chosen is SP800-56B compliant [21], using KTS-OAEP-basic, without key confirmation.

The AES256 has been chosen for encryption and decryption as the algorithm and key length is safe to use and no security risk is currently known (see NIST SP 800 131A [20]). Also, RSA is acceptable, with |n|=2048, for SP800-56B [21] key agreement schemas. Note $|n|$ is the length in bits of the RSA modulus $(n = pq)$, and $|n| = 2048$ means $|n|$ is at least 2048.

AES-256 is a block cipher, being able to encrypt/decrypt messages of a fixed length (called block, in AES it's 128). In order to be able to encrypt/decrypt larger messages (larger than one block size), a mode of operation is required which is an algorithm that describes how to apply the block cipher many times and how to be able to work with larger messages.

Selected mode of operation is Galois Counter Model (GCM), as recommended in "XMLENCR-CORE1 [16]". For details on GCM, see NIST SP 800-38D [27].

The certificate used to encrypt shall be X.509 and shall also be included in the XML file (as allowed by W3C encryption schema [14]) to be able to identify the private key corresponding to this certificate (when decrypting).

Basic steps for decryption are:

— read XML document (W3C Encryption schema);

— extract the RSA certificate to ask for (or look for) the corresponding private key;

— decrypt AES secret key using private key;

— decrypt XML element (xenc:CipherValue) with the encrypted content using the secret key;

— as the content of the decrypted element should be the file, store this file externally in the file system.

6.2.2 File name changes upon encryption

As the table 1 shows, when an encryption is applied to a file that has a reserved extended suffix (or, if there is none, a standard suffix), this reserved extended suffix (or, if there is none, a standard suffix) shall change into .encrypted.xml.

Similarly, when an encryption is applied to a file that has no suffix, the reserved extended suffix .encrypted.xml shall be added to the filename.

Table 1 — Encrypted file name examples

File to encrypt	**Name of the encrypted file**	**Filename inside the XML-enc file**
Lol	Lol**.encrypted.xml**	Same as « File to encrypt »
Lol**.pdf**	Lol**.encrypted.xml**	Same as « File to encrypt »
Lol**.zip**	Lol**.encrypted.xml**	Same as « File to encrypt »
Lol**.signed.xml**	Lol**.encrypted.xml**	Same as « File to encrypt »
Lol**.encrypted.xml**	Lol**.encrypted.xml**	Same as « File to encrypt »

6.2.3 Decrypting a file

This operation is the inverse operation of “Encrypting a file”.

The filename of the decrypted file should become the filename inside the XML signature file.

6.2.4 Signing a file

The present chapter explains the requirements and determines the standard finally chosen for applying electronic signatures.

6.2.5 Requirements

The requirements for the choice of the standard were:

- provide non-repudiation: assure the sender identity, preventing an individual from denying that have effectively signed data;
- prevent the unauthorised (or accidental) modification of data;
- allow the addition of multiple files to a single signature envelope;
- be compliant with European Directive 1999/93/EC [8];
- use a PKI infrastructure, if required;
- shall contain the signer’s digital X.509 certificate;
- shall contain the signing time;
- should include information about policy to verify electronic signature. Hence this signature policy is a legal/contractual document and it might not be available for some authorities. The standard shall support both situations, whether the regulator has a signature policy or not;
- avoid the use of MD5 or SHA-1;
- long term validation is not needed, as signature should be validated in a limited time-frame.

6.2.6 Electronic signature to use

The file structure generated by the signature shall be XAdES-BES/EPES as specified in ETSI-XAdES [2];

The algorithm shall be RSA with SHA512:

http://www.w3.org/2001/04/xmldsig-more#rsa-sha512;

XAdES-BES/EPES (which has been built up on W3C XML Digital Signature) shall be implemented according to COMMISSION DECISION of 25 February 2011 establishing minimum requirements for the cross-border processing of documents signed electronically by competent authorities under Directive 2006/123/EC of the European Parliament and of the Council on services in the internal market [9].

A signature policy is a legal document that extends the definition of the electronic signature by supplementary properties to respect for signature validation. Depending on the availability of such a signature policy, the file structure to generate shall be:

- XAdES-EPES if an explicit signature policy has been defined by the regulator interested in using this standard;
- XAdES-BES if no signature policy has been defined by the regulator for use with this standard.

The digital signature for containers will be "SignatureEnveloping", i.e. the output will be an XML file containing as well the signature as the original file. A ds:object element shall contain the Base64 encoding of the file to be signed (if multiple compressed files are needed in the same signature, multiple ds:object elements may be generated). Attributes MimeType, ID, and Encoding shall be included in the ds:object element. ID should be used to store the file-name to enable regeneration of original filename.

Selected signature algorithm for this standard is RSA with SHA-512 as a hash function. The length of the RSA modulus should be at least 2048 $(|n| \geq 2048)$, a lower value is disallowed (NIST SP 800-131A [20]). Details on RSA can be found in RFC 3447 [28].

The hash function is SHA-512 as specified in FIPS PUB 180-4 [11]. SHA-512 provides a security strength of 256 bits (NIST SP 800-57 part1 [19]).

6.2.7 File name changes upon signature

As the table 2 shows, when a signature is applied to a file that has a reserved extended suffix (or, if there is none, a standard suffix), this reserved extended suffix (or, if there is none, a standard suffix) shall change into .signed.xml.

Similarly, when a signature is applied to a file that has no suffix, the reserved extended suffix .signed.xml shall be added to the filename.

Table 2 — Signed file name examples

File to sign	**Name of the signed file**	**Filename inside the XML signature file**
Lol	Lol.signed.xml	Same as « File to sign »
Lol**.pdf**	Lol.signed.xml	Same as « File to sign »
Lol**.zip**	Lol.signed.xml	Same as « File to sign »
Lol**.signed.xml**	Lol.signed.xml	Same as « File to sign »
Lol**.encrypted.xml**	Lol.signed.xml	Same as « File to sign »

6.2.8 Validating and extracting a signed file

This operation is the inverse operation of “Signing a file”.

The filename of the extracted file shall become the filename inside the XML signature file.

6.3 Creating a submission container

In accordance with the requests from EBA and EIOPA, two main characteristics should be given for the Header that is included in every submission container:

— there should be a basic header, which should be small and easy to use;

— the basic header should be extensible with fields required by the receiver.

These requirements implied a structure of the Header as described in the present chapter.

6.3.1 Header schema structure

The structure of a header as described in this CWA is that of an ExtendedHeader that is to be defined as illustrated in Figure 5. The ExtendedHeader structure shall import the BasicHeader structure and optionally may import other modules like the RegisteredOrganisationVocabulary (continuative work of the « Core Vocabularies » of the EC's Interoperability Solutions for public Administrations programme) and/or other modules (to be developed in the future).

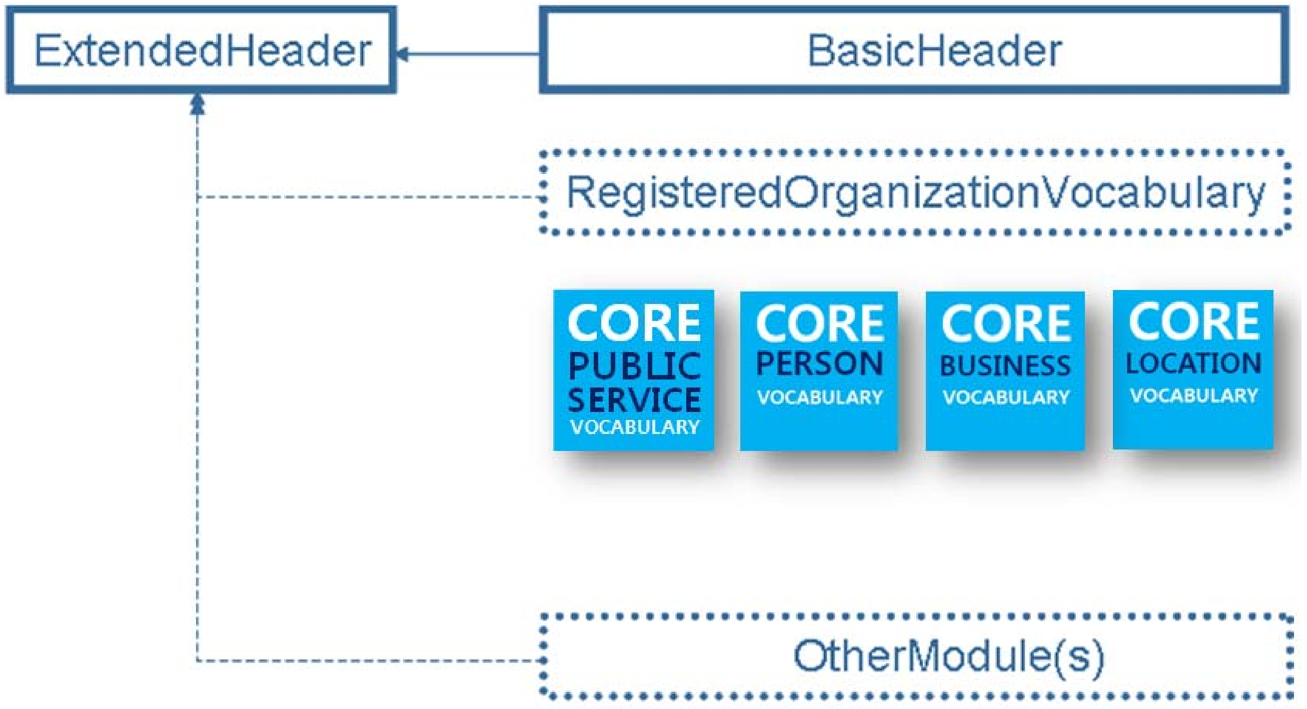


Figure 5 — Extended Header structure importing the BasicHeader structure that optionally imports the RegisteredOrganisationVocabulary (continuative work of the « Core Vocabularies », EC's Interoperability Solutions for Administrations) and/or other modules (to be developed in the future)

The table 3 describes the structure of such a header.

Table 3 — Characteristics of the XML schemas

Header component	**Characteristics**
BasicHeader	This header structure is the « smallest possible » header structure. It consists only of an identifier of the report (set) as well as of the list of data files composing the submitted report (set). This schema shall be imported into any ExtendedHeader.
ExtendedHeader	This header is an adequate header structure that is to be defined by a receiver (that wants to make use of the present CWA) as the header structure to be used by all senders. As an alternative, if no specific requirements regarding header elements exist, one of the pre-defined standard headers defined in the next chapter may be used.
RegisteredOrganizationVocabulary (RegOrg)	On 28th May 2013 the Core Business Vocabulary (EC's Interoperability Solutions for Administrations programme) has been formally published on the W3C standards track as a First Public Working Draft. It has been revised and renamed into Registered Organization Vocabulary [24]. The integration of the RegisteredOrganizationVocabulary into the ExtendedHeader is optional, but it should be imported if the usage of any fields defined in the RegisteredOrganizationVocabulary are required in the ExtendedHeader

Every receiver may thus define an ExtendedHeader structure in accordance with the local needs, giving it an adequate namespace.

6.3.2 Predefined standard use-cases of ExtendedHeader schema

The following use-cases, presented in table 4, for creating an ExtendedHeader are explicitly defined by the present CWA and may be used « as is ».

Table 4 — 6.3.2 Predefined standard use-cases for extended headers

ExtendedHeader - pre-defined use-case	**Characteristics**
BasicHeaderOnly	This header imports the BasicHeader « as is », makes no extensions for it and does not import the RegisteredOrganizationVocabulary as it doesn't use any of its fields. **Namespace:** http://www.eurofiling.info/eu/fr/esrs/Header/BasicHeaderOnly **XSD URL:** http://www.eurofiling.info/eu/fr/esrs/Header/BasicHeaderOnly.xsd **XML sample instance URL:** http://www.eurofiling.info/eu/fr/esrs/Header/BasicHeaderOnly.xml

StandardHeader**With**RegOrg	This header structure reflects the survey made within the Eurofiling BestPractices [25]. All fields related to « Transport » issues have been removed as these are out of scope of this CWA. **Namespace:** http://www.eurofiling.info/eu/fr/esrs/Header/StandardHeader**With**RegOrg **XSD URL:** http://www.eurofiling.info/eu/fr/esrs/Header/StandardHeader**With**RegOrg.xsd **XML sample instance URL:** http://www.eurofiling.info/eu/fr/esrs/Header/StandardHeader**With**RegOrg.xml
StandardHeader**Without**RegOrg	This header structure is (with regards to its function and its content) equivalent to the previous “StandardHeader**With**RegOrg”, but it does not import RegOrg and creates the missing fields as equivalent simple XML fields. **Namespace:** http://www.eurofiling.info/eu/fr/esrs/Header/StandardHeader**Without**RegOrg **XSD URL:** http://www.eurofiling.info/eu/fr/esrs/Header/StandardHeader**Without**RegOrg.xsd **XML sample instance URL:** http://www.eurofiling.info/eu/fr/esrs/Header/StandardHeader**Without**RegOrg.xml

6.3.3 Creating a specific ExtendedHeader schema

The guidelines for the creation of a specific ExtendedHeader schema are given in Annex G.

6.3.4 Creating a header file

The creation of a header file consists of the actions:

— assembling the data files;

— creation of the header file according to the ExtendedHeader XML schema chosen (as defined by the receiver), the BasicHeader part of the ExtendedHeader listing the assembled data files

6.4 Creating a response container

The creation of a response container consists of the actions presented in the following paragraphs.

6.4.1 Creating a container feedback file

The creation of a container feedback file should take place in accordance with the documentation of the ContainerFeedback schema in Annex D.

6.4.2 Creating Instance feedback (Validation, usually only for XBRL)

The creation of an instance feedback file should take place in accordance with the documentation of the InstanceFeedback schema in Annex E.

7 Exchange model

This chapter will introduce the exchange model to be used among a sender and a receiver. The receiver should emit instructions on how to use the present CWA for the given exchange of information between a sender and the receiver.

The exchange of information composes of the phases presented in the following paragraphs.

7.1 Phase 1: the sender creates a submission container, applies all security mechanisms required and transmits it to the receiver

The sender makes use of the adequate transport mechanism to submit the container with the data instances.

7.2 Phase 2: the receiver processes the security layer(s) on the container and all the files within

The receiver removes all encryption layers and verifies all signatures as indicated by the reserved extended suffixes on the container and the files there-in.

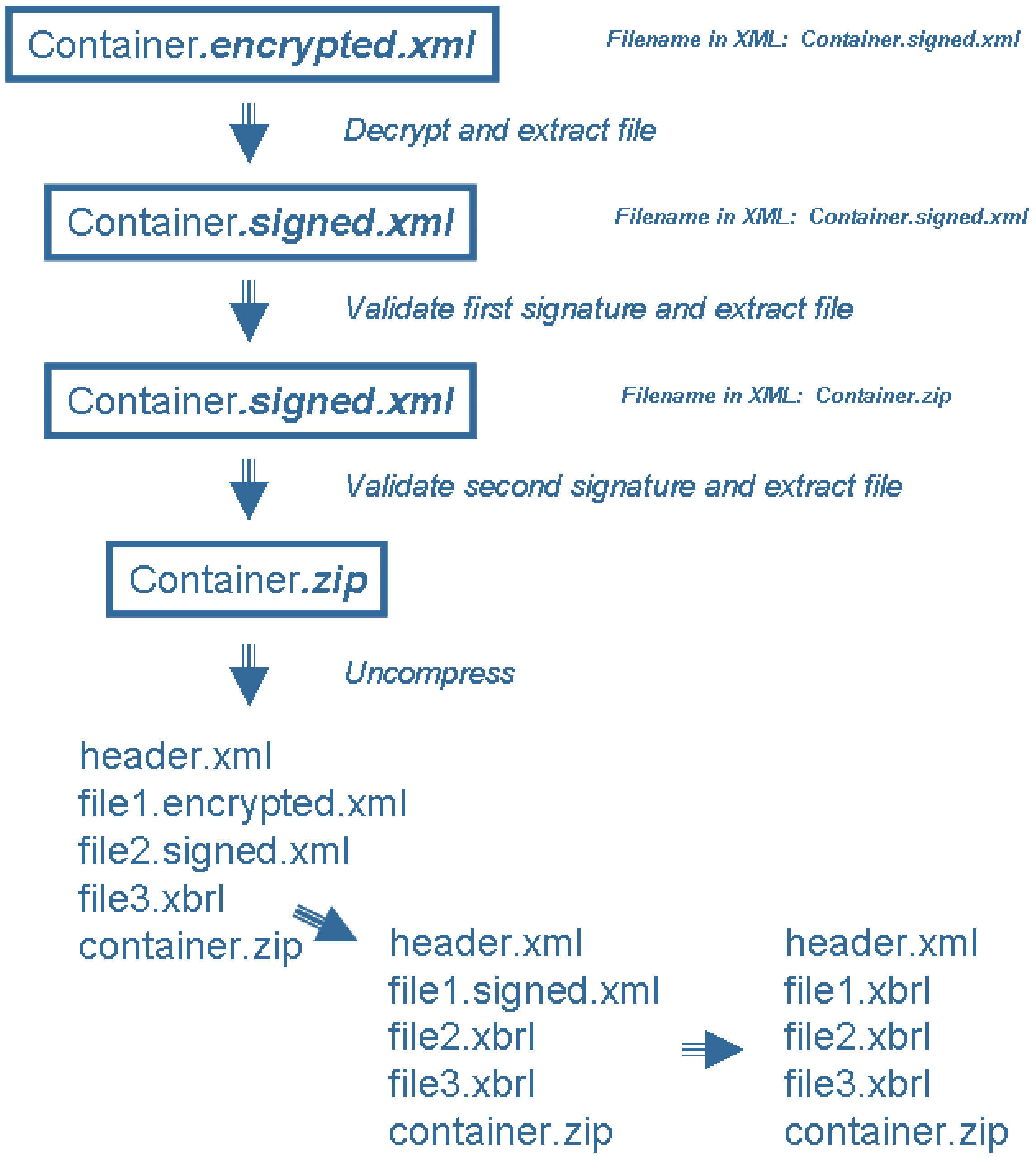


Figure 6 — Illustration of the security removal at container level (in this example two signatures and one encryption have been applied) as well as on all the files within the container

The container should be reviewed to make sure it has the correct structure (does the header file validate correctly, are all the files announced in the header effectively part of the container, etc.).

7.3 Phase 3: receiver generates a positive / negative acknowledgement for the reception of the submission container

As a result of the processing in the preceding phase, it is now identified if the container could be correctly received or if it was invalid (container or files within not decryptable, signature(s) invalid, entity not known or not authenticable, decompression failed, etc.). A container feedback file should be generated that either confirms the validity of all reception steps (like security removal and supplementary checks) and results in a positive acknowledge, or that lists (in the opposite case) the errors that occurred (negative acknowledgement).

The acknowledgement will be included into the response container in phase 5

7.4 Phase 4: the receiver processes the contents of the container

In case of a positive acknowledgement, the data files submitted shall go through the following stages of processing.

The standard suffixes of the files shall be used to identify those files that can be further processed. As an example, XBRL or XML instances should now be validated by their respective validator (while unstructured data files like word processor files could be made available to analysts for manual review).

As a result of this phase, the adequate (alternative) instance feedback files for the XBRL instances in the original submission container should have been generated.

7.5 Phase 5 (optional): the receiver returns the validation result of the data files in the response container

All feedback files as well as the container feedback file may be added to a response container that should be returned by the receiver to the sender to provide the result of the content processing of the related submission container.

Unlike the submission container, a response container shall not include header information.

The receiver may alternatively make available the result of the processing in a way considered more appropriate (e.g. returning links to external systems etc.).

The exchange model can thus be presented as in figure 7.

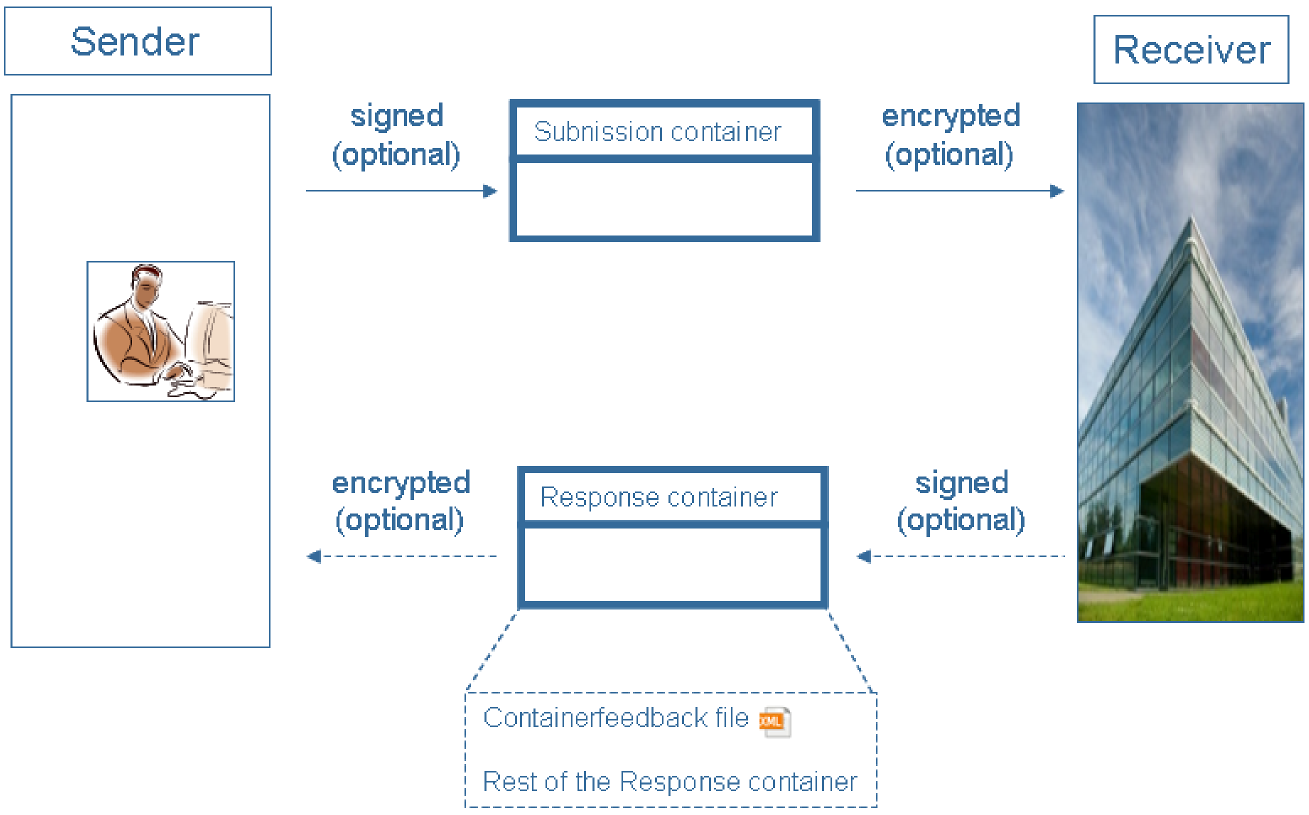


Figure 7 — Illustration of the exchange model

Annex A
(normative)

Items that shall be defined in the instructions

A.1 Introduction

These are some questions to which any institution willing to use the CWA has to give clear answers to in its instructions.

A.2 Container structure

Table A.A.1 — Common questions and instructions for the container

Item	Typical instruction	Explanations
Use of CWA encryption layer?	YES, one single encryption to be applied on a signed file (container)	should be Yes, may be No in environments using a secure transport
Use of CWA signature layer?	YES, one single signature to be applied on a zip compressed file (container)	should be Yes, may be No in environments using a secure transport
Use of the « Instancefeedback » schema» to inform of the result of the processing of a data instance?	« Use instance feedback to report errors » or « Use Excel files in folder XBRL_Errors instead »	« None » or « InstanceFeedback schema » or « explanation of an alternative mechanism (e.g. provision of links to an external system of graphical representations of error conditions) »

A.3 Header

These are required precisions on how some tags of the header schema shall be used. These instructions are only required if a standard header schema is used; otherwise the according fields could simply be omitted in a customised header extension.

Table A.A.2 — Common instructions for the header

Item	**Typical instruction**	**Explanations**
<ReportReferenceID>	e.g. « Finrep full quarterly consolidated reporting for investment companies » or a code for that reporting	The list of the different reporting identifications covered by the instructions
<AuditStatus>	e.g. « The values « audited » and « not audited » shall be used exclusively »	Either confirm the use of the flag or specify that the value « undetermined » or « in datainstances » is applicable
<ConsolidationStatus>	See <AuditStatus>	See <AuditStatus>
<CapitalCurrency>	Capital Currency shall be used and be EUR mandatorily	This tag may be used for validation purposes in some countries imposing the exclusive use of a single currency
<UpdateStatus>	e.g. « Only use value « Replace » »	If no « update » mechanism is provided, the value « Replace » should be enforced by the instructions, indicating that all prior reports of the same type will be deleted and replaced by the content of the new container In the other case, allowing the values « Update » (keep any values from a prior reporting excepted for those in the data instances of the present container which will be replaced by their new values) and « Delete » (delete any values from prior reports that are in the data instances of the present container) can make sense
<TestFlag>	« All data are production data, do not use <TestFlag>true</TestFlag> »	This tag may be used to flag data instances sent to the receiver's production infrastructure as test data (to be validated, but not injected into the receiver's databases)

A.4 Lists of codes accepted

A table like table A.3 (defining identifiers accepted both for legal entities and for persons) should explain which codes are allowed.

Table A.A.3 —Identifiers accepted both for legal entities and for persons

Code type	**Issuer**	**Country**	**URI**
<IdentifierType>	<IdentifierIssuingAuthority>	<IssuingAuthorityCountry>	<IssuingAuthorityURI>

Annex B
(informative)

Supplementary items that may be useful in the instructions

Table B.B.1 — Supplementary items and explanations

Item	**Explanations**
Name of the use of the CWA	Examples: « Prudential supervisory reporting », « XBRL reporting only », « NSA to EBA transmissions »
Applicable File naming conventions (if any)	E.g. a link to an external document describing all applicable file naming conventions applicable to containers, folders, data instances, unstructured files
Containers for other regulators to include	Description and destination authorities for the containers etc.
Supplementary rules	Example of a supplementary rule: a certain container inside the container shall contain the same instances (reported instances for second level reporting are the same as for first level reporting)

Annex C
(informative)

Explanations on header schema

The present specification allows extensible headers which makes it difficult to choose a certain header to document. For illustration purposes, the present chapter documents the standard extensible header "StandardHeaderWithoutRegOrg".

The XML Schema StandardHeaderWithoutRegOrg is depicted in figures C.1 to C.4.

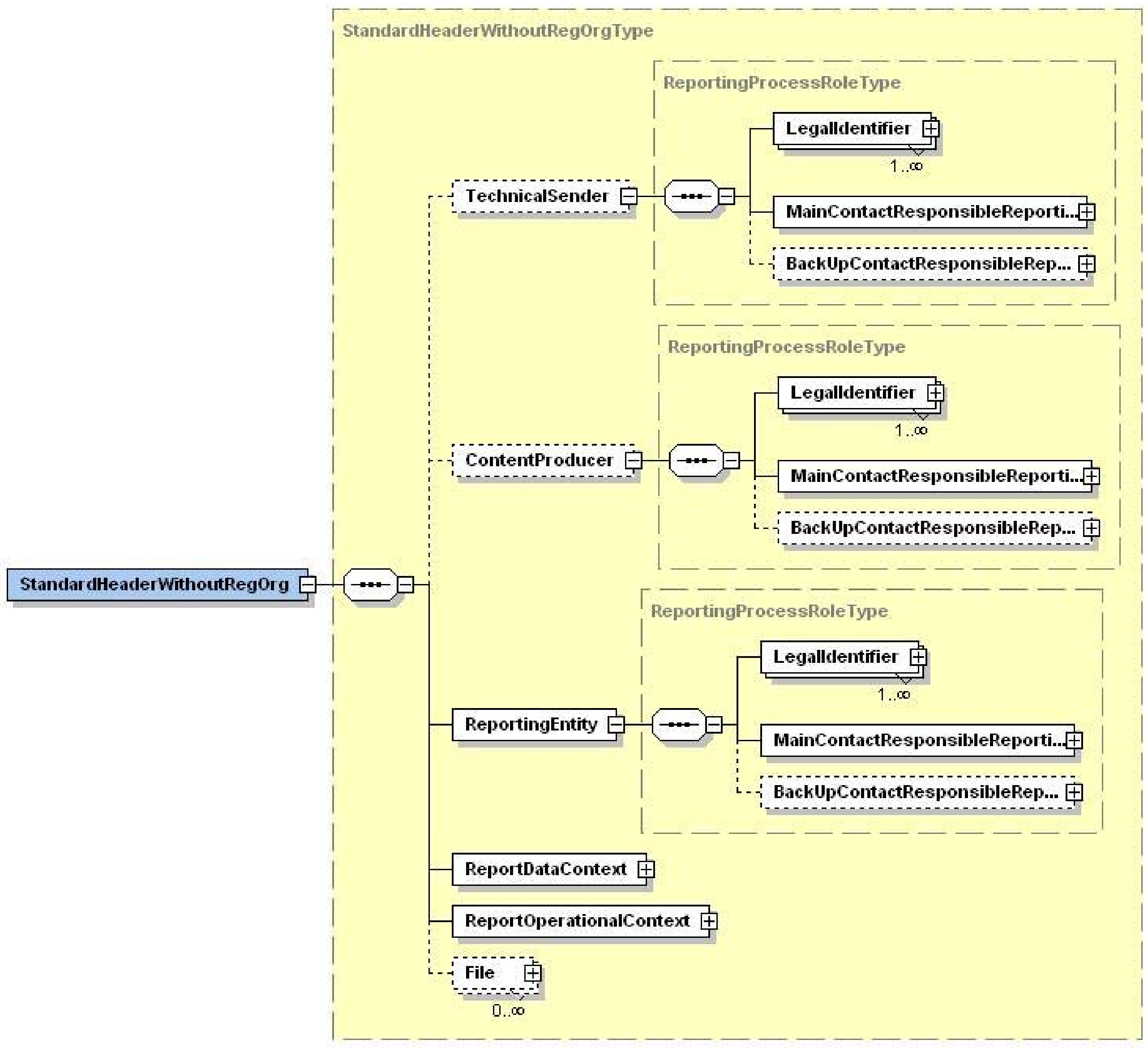

StandardHeaderWithoutRegOrgType
StandardHeaderWithoutRegOrg
TechnicalSender
ContentProducer
ReportingEntity
ReportDataContext
ReportOperationalContext
File
0..∞
ReportingProcessRoleType
LegalIdentifier
1..∞
MainContactResponsibleReporti...
BackUpContactResponsibleRep...
ReportingProcessRoleType
LegalIdentifier
1..∞
MainContactResponsibleReporti...
BackUpContactResponsibleRep...
ReportingProcessRoleType
LegalIdentifier
1..∞
MainContactResponsibleReporti...
BackUpContactResponsibleRep...

Figure C.C.1 — StandardHeaderWithoutRegOrg (TechnicalSender, ContentProducer and ReportingEntity)

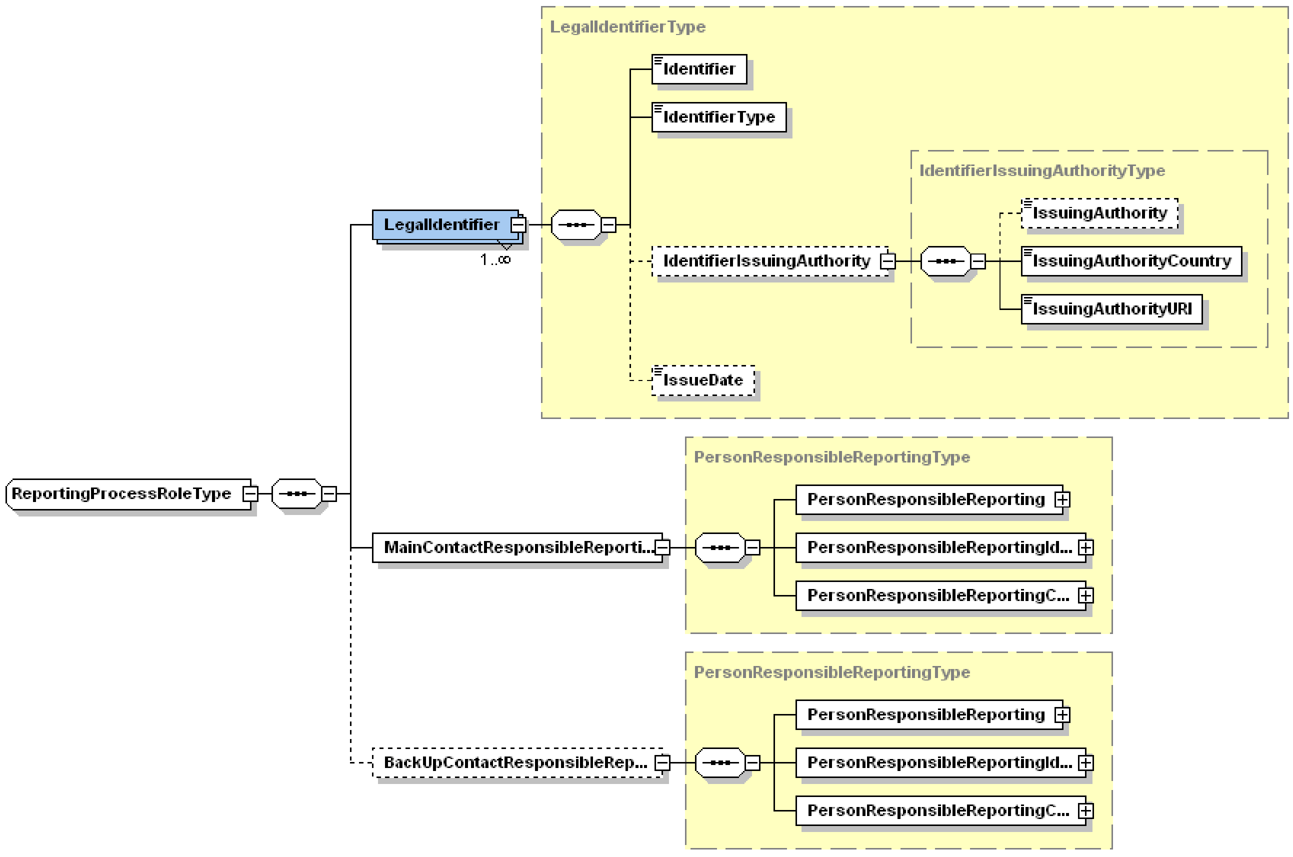

LegalIdentifierType
Identifier
IdentifierType
IdentifierIssuingAuthorityType
IssuingAuthority
IssuingAuthorityCountry
IssuingAuthorityURI
LegalIdentifier
1..∞
IdentifierIssuingAuthority
IssueDate
ReportingProcessRoleType
PersonResponsibleReportingType
PersonResponsibleReporting
MainContactResponsibleReporti...
PersonResponsibleReportingId...
PersonResponsibleReportingC...
PersonResponsibleReportingType
PersonResponsibleReporting
BackUpContactResponsibleRep...
PersonResponsibleReportingId...
PersonResponsibleReportingC...

Figure C.C.2 — ReportingProcessRoleType

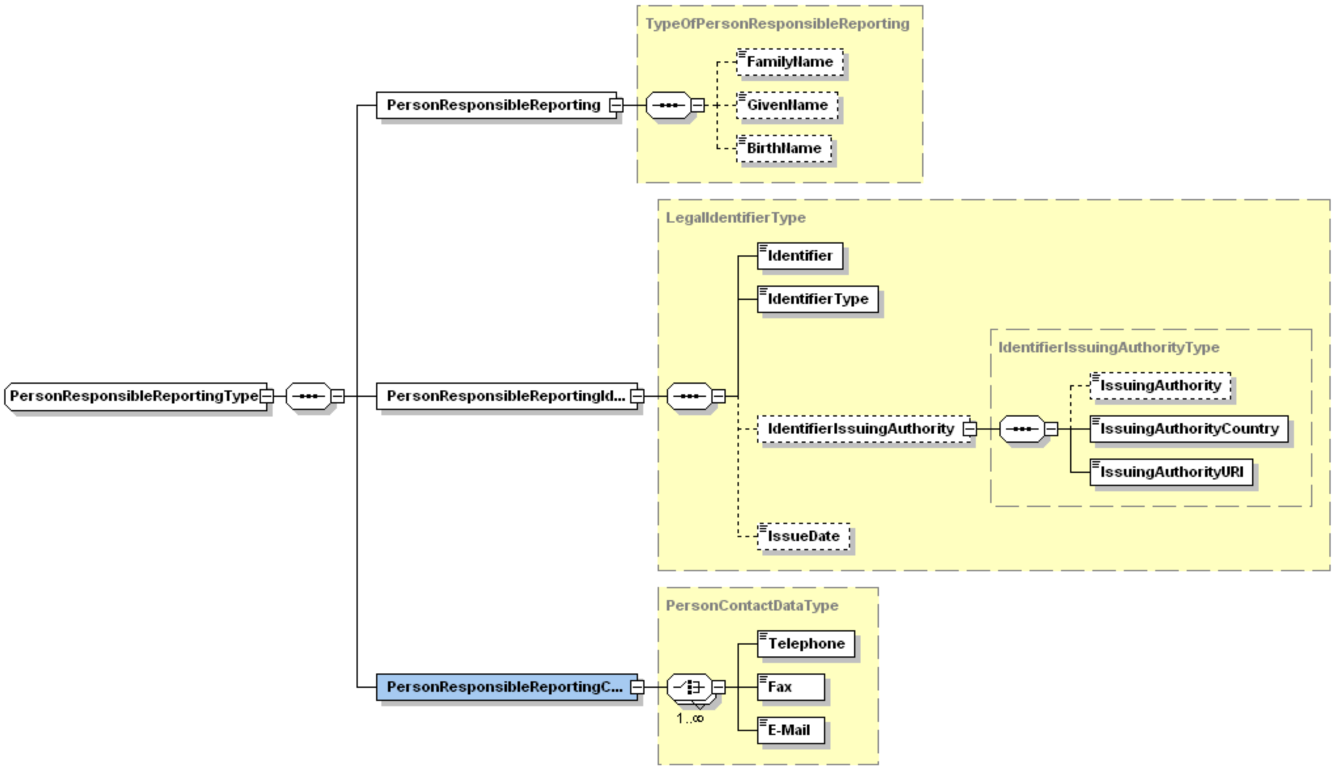

TypeOfPersonResponsibleReporting
FamilyName
GivenName
BirthName
PersonResponsibleReporting
LegalIdentifierType
Identifier
IdentifierType
IdentifierIssuingAuthorityType
IssuingAuthority
IssuingAuthorityCountry
IssuingAuthorityURI
IdentifierIssuingAuthority
IssueDate
PersonResponsibleReportingType
PersonResponsibleReportingId...
PersonContactDataType
Telephone
Fax
E-Mail
PersonResponsibleReportingC...
1..∞

Figure C.C.3 — PersonResponsibleReportingType

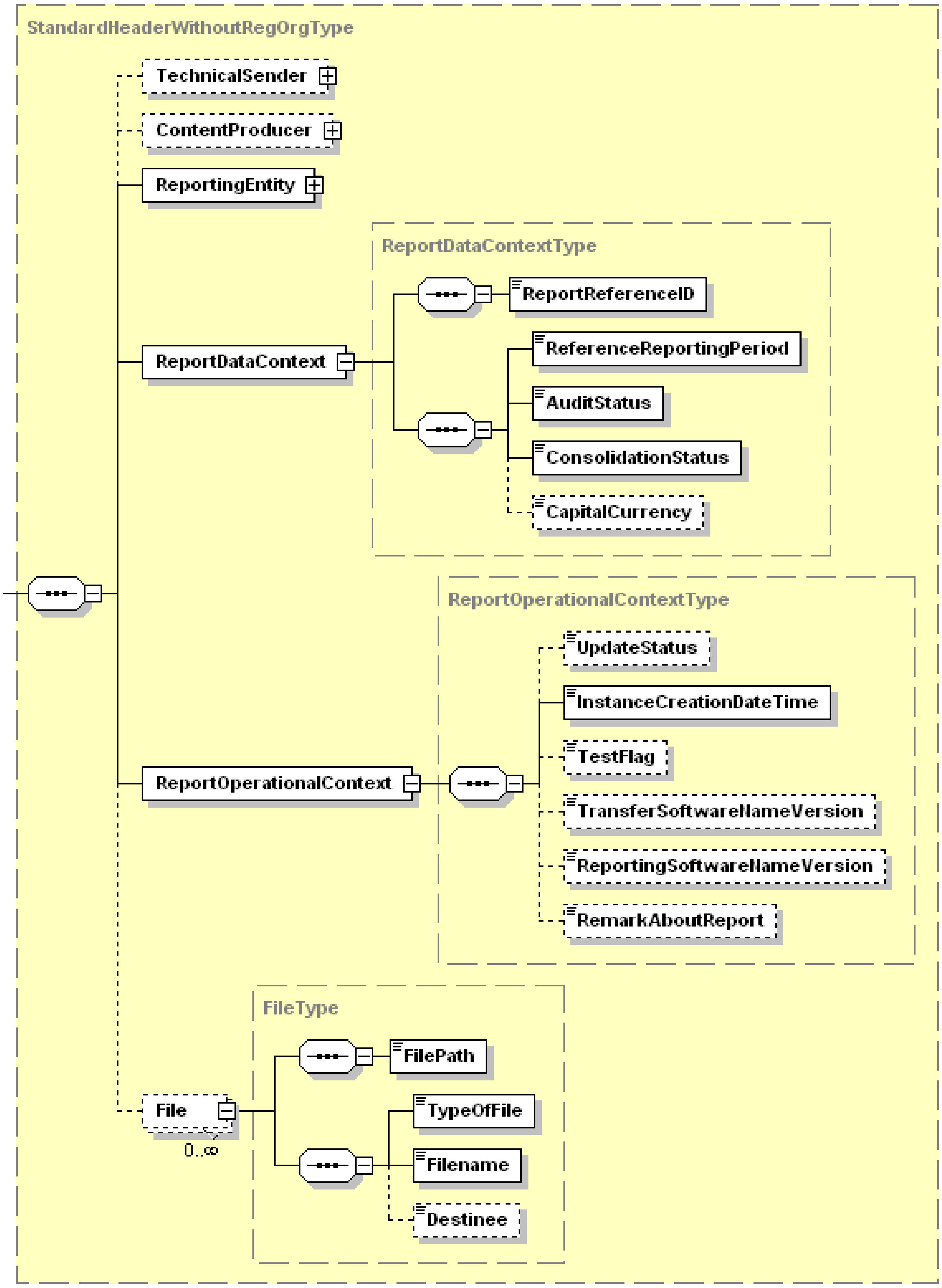


Figure C.C.4 — StandardHeaderWithoutRegOrg (ReportingDataContext, ReportOperationalContext and File)

Table C.C.1 — StandardHeaderWithoutRegOrg explanations

Item	**Type**	**Explanation**
ReportDataContext	ReportDataContextType	
File	FileType	
TechnicalSender	ReportingProcessRoleType	(Potential) sub-contractor in charge of physically sending the data in respect of the present CWA (aware of containers, encryption, etc.).
ContentProducer	ReportingProcessRoleType	(Potential) sub-contractor in charge of the production of the content of the reporting and responsible for the accuracy of the content.
ReportingEntity	ReportingProcessRoleType	Entity submitted to financial reporting and legally responsible for it (in many cases it uses internal resources to play the role of Content Producer and Technical Sender too). An authority may also play the role of a reporting entity, e.g. when a national authority is providing data to a subsequent European authority as level-2 reporting.
ReportDataContext	ReportDataContextType	Sequence of information that defines the general context of the data in the report i.e. general, common properties applicable to all the reporting files
ReportOperationalContext	ReportOperationalContextTyp e	Sequence of information that defines properties related to the process of submitting data
FileType	sequence	
FilePath	relative URI	This field gives the relative Uniform Resource Identifier (URI) to a file in the container (starting from top-level).
TypeOfFile	text enumeration	Potential supplementary file type characterizing each individual file in the container to allow supplementary dedicated processing based of the file type. Possible values: "DataInstance", "OtherFile", "SignedAndEncryptedSubcontainer", "SignedSubcontainer", "CompressedOnlySubcontainer"
Filename	text	Explicit name of the file
Destinee	identifier	Potential destinee for one of the files in the container, e.g. a container inside a container
ReportingProcessRoleTy pe	sequence	
LegalIdentifier	LegalIdentifierType	Sequence of properties identifying the reporting entity
MainContactResponsibleR eporting	PersonResponsibleReportingT ype	Human contact for the case of problems with a certain step of the reporting generation and submission process
BackUpContactResponsibl	PersonResponsibleReportingT	Backup person for BackUpContactResponsibleReporting

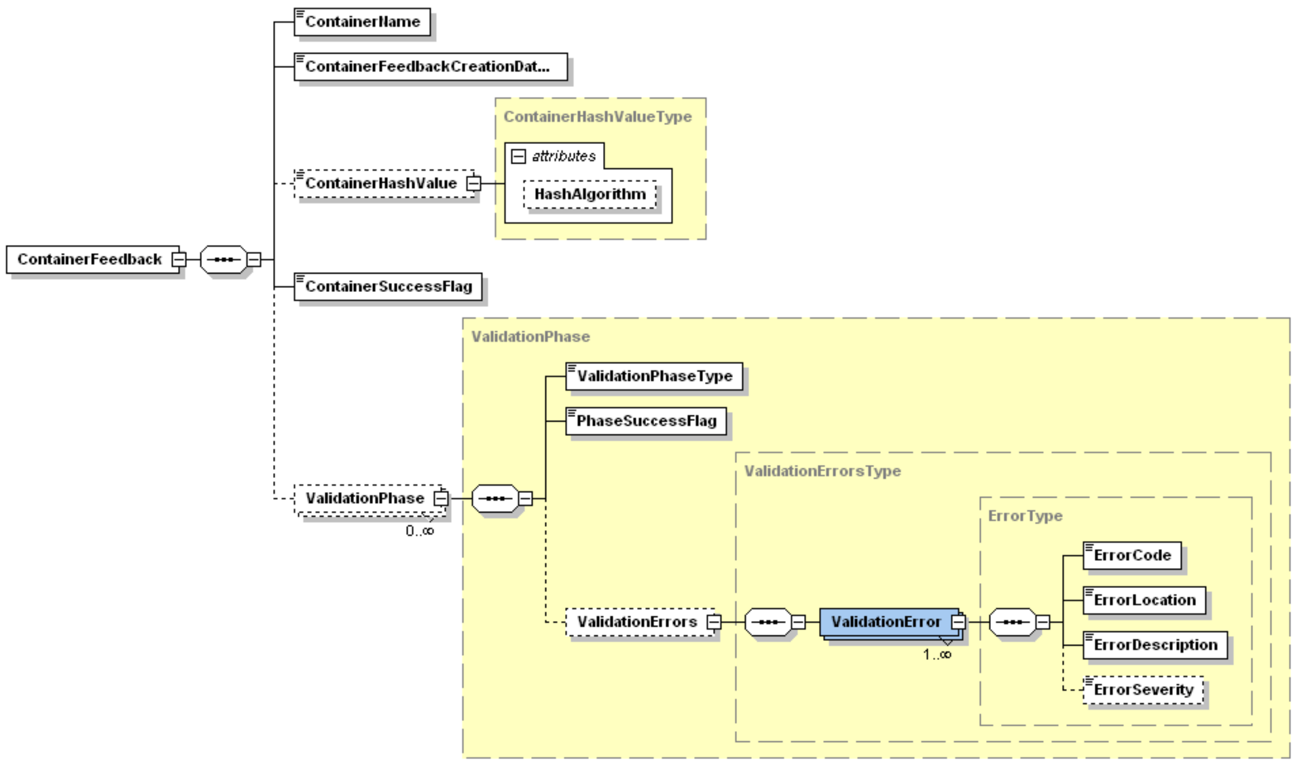
eReporting	ype	
LegalIdentifierType	sequence	
Identifier	identifier	Code identifying the reporting entity
IdentifierType	code	Type of code identifying the reporting entity
IdentifierIssuingAuthority	IdentifierIssuingAuthorityType	Sequence of information that describes the authority having issued the certificate
IssueDate	date	Issuing date of the code
IdentifierIssuingAuthorityType	sequence	
IssuingAuthority	text	Name / Identifier of the issuing authority
IssuingAuthorityCountry	code	ISO country code of the issuing authority
IssuingAuthorityURI	URI	URI identifying the issuing authority
PersonResponsibleReportingType	sequence	
PersonResponsibleReporting	TypeOfPersonResponsibleReporting	Sequence of information describing the person that is in charge of the reporting
PersonResponsibleReportingIdentifier	LegalIdentifierType	Sequence of information describing the Identifier of the person that is in charge of the reporting
PersonResponsibleReportingContactData	PersonContactDataType	Sequence of information describing means to contact the person that is in charge of the reporting
TypeOfPersonResponsibleReporting	sequence	
FamilyName	text	person family name
GivenName	text	person given name
BirthName	text	person birth name
PersonContactDataType	choice element	Choices available: Telephone, Fax, E-Mail
ReportDataContextType	sequence	
ReportReferenceID	identifier	This code identifies the data submitted in the container. It can be a set of reports (e.g. code “FINREP_COREP” for Finrep & Corep), a single report (e.g. “QUARTERLY_CONSOLIDATED_FINREP” for the standard Finrep) or a subset of reports (e.g. “TABLE1&2 FINREP” for only the according subset of Finrep)
ReferenceReportingPeriod	date	Main reporting period (end date of the period)

AuditStatus	text enumeration	Data extracted from general ledger ("not audited") or having undergone an external audit already ("audited")
ConsolidationStatus	text enumeration	Consolidated or solo in different flavours. Possible values: "solo head office excluding branches", "solo head office including branches", "solo branch only", "sub-consolidated", "consolidated"
CapitalCurrency	code	Main currency
ReportOperationalContextType	sequence	
UpdateStatus	text enumeration	Flag characterizing if this is an entirely new report ("Replace") or if it is an update of a previously sent report ("Update") or if the prior report should be deleted ("Delete")
InstanceCreationDateTime	date	Creation date & time of the instance
TestFlag	Boolean	Flag to characterize if it is actual production data (false, default value) or only test data (true)
TransferSoftwareNameVersion	text	Software or system used to submit the report
ReportingSoftwareNameVersion	text	Software or system used to generate the report
RemarkAboutReport	text	Remark on the report

Annex D
(informative)

Documentation of the container feedback schema

The present chapter describes the XML Schema ContainerFeedback that is depicted in figure D.1.



D.

Figure D.D.1 — Visualisation of the container feedback schema (ContainerFeedback.xsd)

In table D.1, there is supplementary information on the schema describing:

— type: the type of the element in the schema and

— description and usage: a narrative explanation for the elements and a recommendation for the scenario in which the element should be used.

Table D.D.1 — Container feedback schema element listing and description

Element name	**Type**	**Description and usage**
ContainerFeedback		The root element, validation consists of separate validation sections
ContainerName	text	The name of the container that has been received by an authority and for which the container feedback document acknowledging successful / unsuccessful reception
ContainerFeedbackCreation DateTime	dateTime	The creation date and time of the container feedback document
ContainerHashValue	ContainerHashValueType	Hash calculated according to the present specification for the container received in order to assure that both sides make reference to exactly the same file and can verify the file integrity.
ContainerSuccessFlag	boolean	Overall view if the container structure as a whole was correct true = container ok false = errors found in at least 1 phase of the reception of the container
ValidationPhase	ValidationPhase	Usage: only when ContainerSuccessFlag is set "false".
ValidationPhase	sequence	
ValidationPhaseType	text	The type of validation phase for instance "decryption" or "decompression".
PhaseSuccessFlag	boolean	true = validation phase successful false = errors found in the validation phase
ValidationErrors	ValidationErrorsType	Lists all the errors found in the validation. Usage: only when PhaseSuccessFlag is set to "false".
ValidationErrorsType	sequence	
ValidationError	ErrorType	The error found
ErrorType	sequence	a generic error type that can be used in all validation sections to define errors found
ErrorCode	code	an error code that can be used to identify the error found
ErrorLocation	text	an expression that can be used to locate the error in the instance document, can be an Xpath sentence or line number
ErrorDescription	text	a description of the error found

ErrorSeverity	ErrorSeverityType	The severity of the error
ErrorSeverityType	text enumeration	Possible values: "Info", "Warning", "Error" and "Fatal"
ValidationPhaseType	text	A description of the validation phase for instance decryption, signature verification, authentication, …
ContainerHashValueType	hash/digest	Has an attribute HashAlgorithm with a fixed value "http://www.w3.org/2001/04/xmlenc#sha256 "

Annex E
(informative)

Documentation of the instance feedback schema

The present chapter describes the XML Schema InstanceFeedback that is depicted in figure E.1.

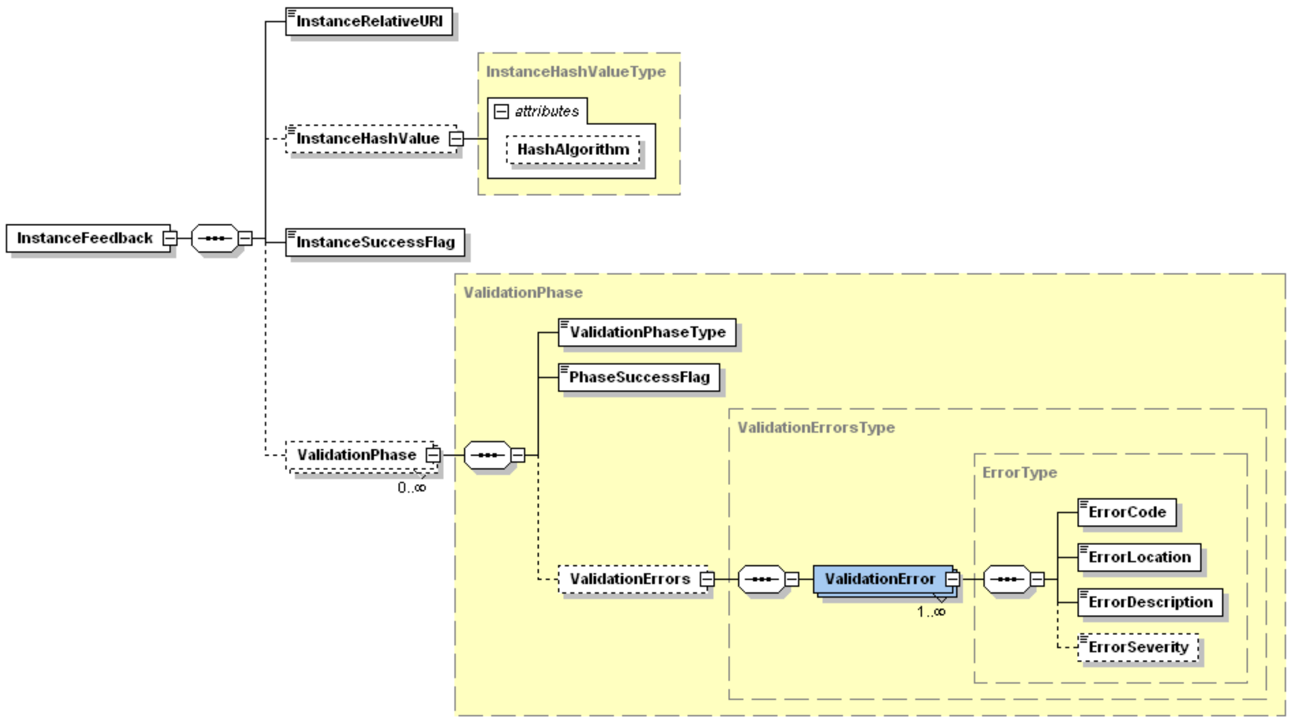


Figure E.E.1 — Visualisation of the instance feedback schema (InstanceFeedback.xsd)

In table E.1, there is supplementary information on the schema describing:

— type: the type of the element in the schema and

— description and usage: a narrative explanation for the elements and a recommendation for the scenario in which the element should be used.

Table E.E.1 — Instance feedback schema element listing and description

Element name	Type	Description and usage
InstanceFeedback		The root element, validation consists of separate validation sections
InstanceRelativeURI	relative URI	shall contain the path to the data instance from the top-level of the submission package in relative URI notation
InstanceHashValue	InstanceHashValueType	Reference to the calculated hash value of the instance document being validated.
InstanceSuccessFlag	boolean	Overall view of all instance validations true = all validations successful false = errors found in at least 1 validation phase
ValidationPhase	ValidationPhase	Any validation: XML, XBRL 2.1 Conformance Suite 1.0 validation, taxonomy validation. Usage: Only when InstanceSuccessFlag is set "false".
ValidationPhase	sequence	
ValidationPhaseType	string	The type of validation phase
PhaseSuccessFlag	boolean	true = validation phase successful false = errors found in the validation phase
ValidationErrors	ValidationErrorsType	Lists all the errors found in the validation. Usage: only when PhaseSuccessFlag is set to "false".
ValidationErrorsType	sequence	
ValidationError	ErrorType	The error found
ErrorType	sequence	a generic error type that can be used in all validation sections to define errors found
ErrorCode	code	an error code that can be used to identify the error found
ErrorLocation	text	an expression that can be used to locate the error in the instance document, can be an Xpath sentence or line number
ErrorDescription	text	a description of the error found
ErrorSeverity	ErrorSeverityType	The severity of the error
ErrorSeverityType	text enumeration	Possible values: "Info", "Warning", "Error" and "Fatal"
InstanceHashValueType	hash value	Has an attribute HashAlgorithm with a fixed value "http://www.w3.org/2001/04/xmlenc#sha256 "

Annex F
(informative)

Guidelines on how to extend the basic header

Step 1

Create your own XSD file, replace the default namespace (http://www.eurofiling.info/eu/fr/esrs/Header/ExtendedBasicHeader) by your own namespace and import the basic header:

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

<xsd:schema xmlns="http://www.eurofiling.info/eu/fr/esrs/Header/ExtendedBasicHeader"
xmlns:bh="http://www.eurofiling.info/eu/fr/esrs/Header/BasicHeader" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.eurofiling.info/eu/fr/esrs/Header/ExtendedBasicHeader" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1">

<xsd:import namespace="http://www.eurofiling.info/eu/fr/esrs/Header/BasicHeader"
schemaLocation="http://www.eurofiling.info/eu/fr/esrs/Header/BasicHeader.xsd"/>
```

Step 2

Create our own elements as an extension of the basic header, for example:

```
<xsd:element name="MyExtendedHeader" type="MyExtendedHeaderType"/>

    <xsd:complexType name="MyExtendedHeaderType">
        <xsd:sequence>
            <!-- My new element -->
            <xsd:element name="MyNewElement" type="xsd:string"/>

            <!-- Basic Header elements -->
            <xsd:element ref="bh:BasicHeader" maxOccurs="1"/>

        </xsd:sequence>
    </xsd:complexType>
```

The basic header elements should be used at the end of the extended schema, and they should only be used once.

Annex G
(informative)

Use cases for this CWA

G.1 Reporting entity to supervisor (1st level)

In this use-case, the sender is the reporting entity, the receiver is the supervisor.

The security mechanisms applied to submission containers should be the same (and have the same order of application) as those applied to response containers.

G.2 Reporting entity to National Supervision Authority (NSA) to European Supervision Authority (ESA) (1st and 2nd level)

In this case, the exchange model is used twice in a row with:

1) exchange 1: the sender is the reporting entity, the receiver is the NSA;

2) exchange 2: the sender is the NSA, the receiver is the ESA.

G.2.1 2-layer submission process with forwarding of information

The NSA requires not only data for its own purpose, but also data in a separate container inside the original container in order to be able to forward this data to a subsequent regulator like an ESA. As a consequence, the ESA needs to know all the public key / certificate of the reporting entities (from which data are sent) as communication partners. Figure H.1 shows a 2-layer submission using containers to forward data to subsequent authorities as well as feedback to the respective sender.

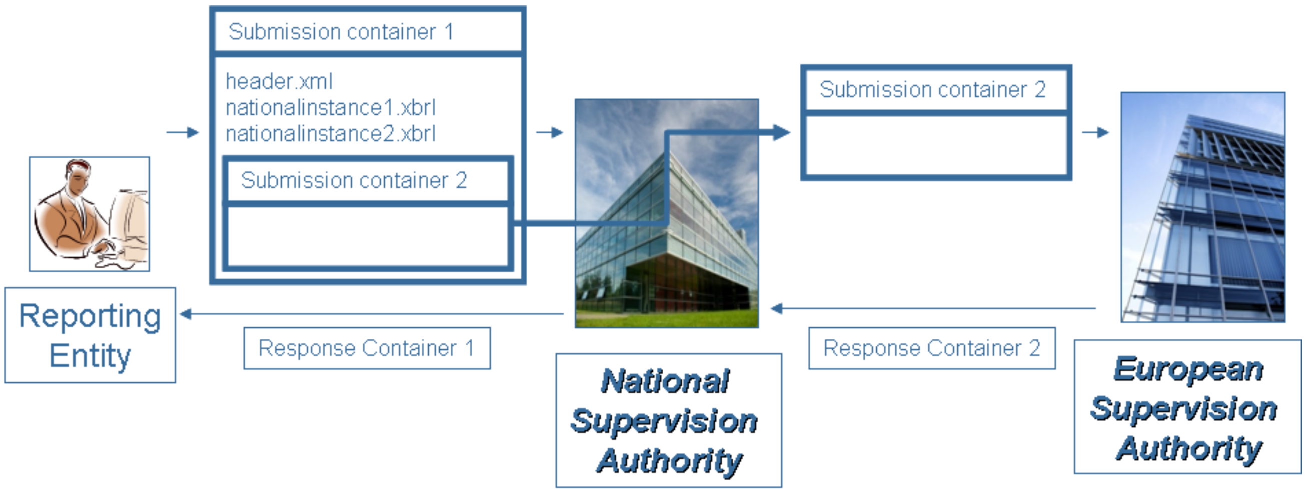


Figure GH.H.1 — 2-layer submission using containers in containers to forward data to subsequent authorities and as well as feedback to the respective sender

G.2.2 2-layer submission process with repackaging or regeneration

After finishing the submission process from the reporting entity to the NSA, a separate, entirely independent submission process is started using either the original data from the entity (repackaging) or entirely new data prepared by the NSA (open regeneration). The ESA has only one communication partner, the NSA, of which it needs to know the public key / certificate. These approaches are illustrated in figures H.2 and H.3.

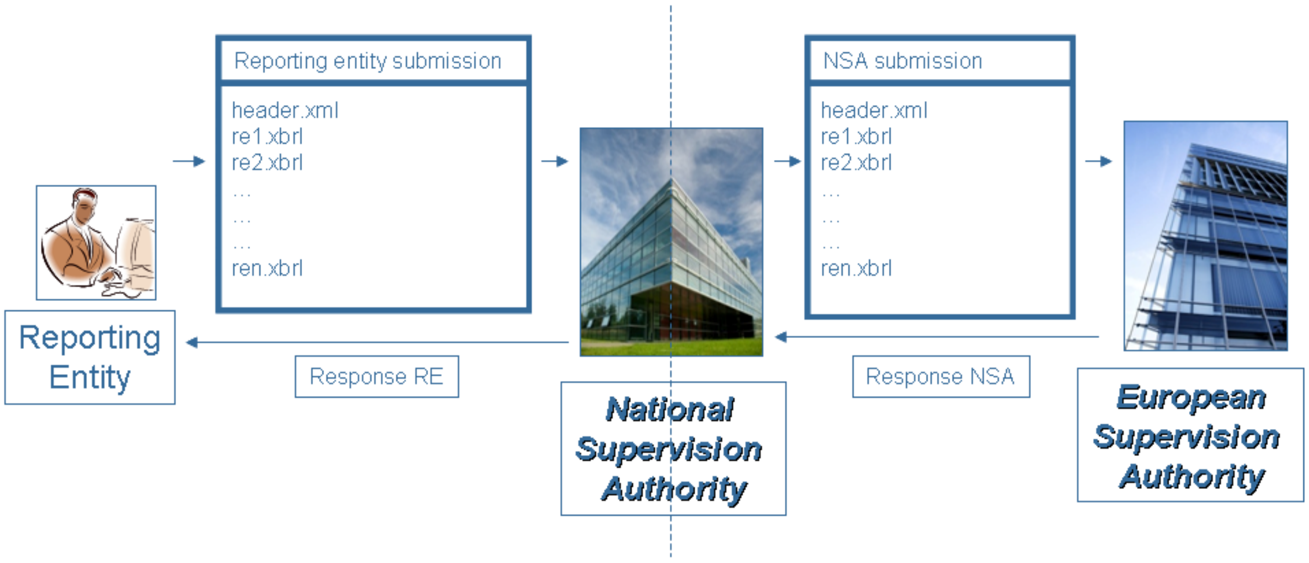


Figure GH.H.2 — 2-layer submission repackaging data into new containers to send them to subsequent authorities

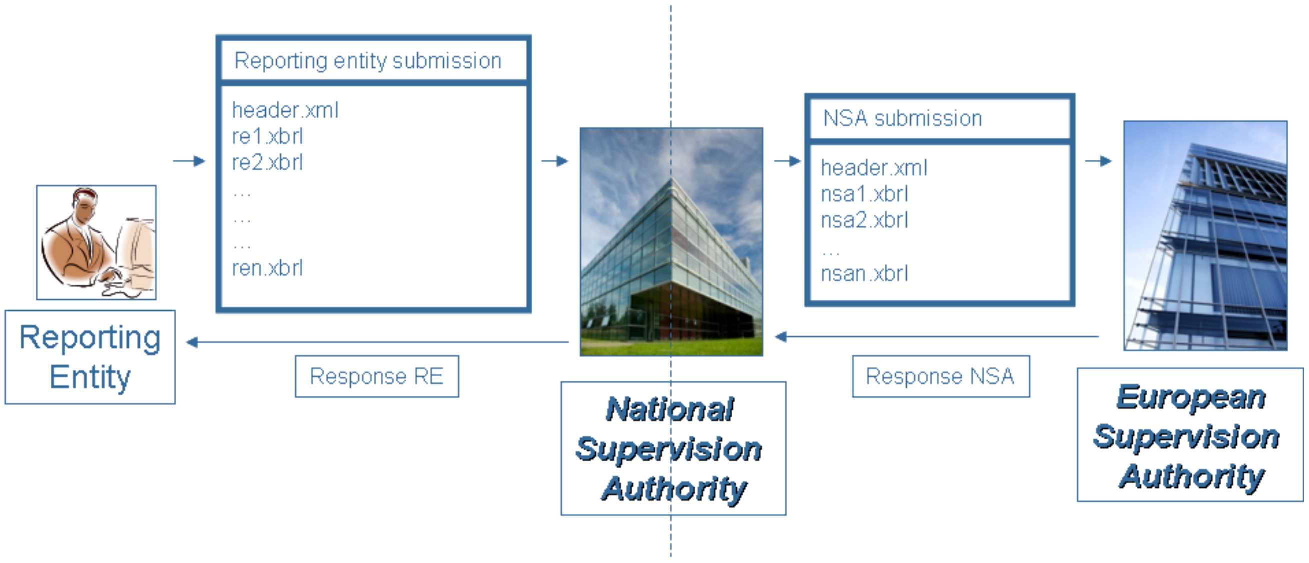


Figure GH.H.3 — 2-layer submission, completely new generation of data from NSA systems for subsequent authorities

Bibliography

[1] ETSI Technical Report 102 038 v1.1.1. Electronic Signatures and Infrastructures; XML format for signature policies. European Telecommunications Standards Institute. April 2004. http://docbox.etsi.org/EC_Files/EC_Files/tr_102038v010101p.pdf

[2] ETSI-XAdES, ETSI Technical Specification 101 903 V1.4.1. XML Advanced Electronic Signatures (XAdES). June 2009. European Telecommunications Standards Institute. http://uri.etsi.org/01903/v1.4.1/

[3] ETSI Technical Specification 102 176-1 v2.0.0, Electronic Signatures and Infrastructures; Algorithms and Parameters for Secure Electronic Signatures; Part 1 Hash functions and asymmetric algorithms.19 November 2007. European Telecommunications Standards Institute. http://www.etsi.org/deliver/etsi_ts/102100_102199/10217601/02.00.00_60/ts_10217601v020000p.pdf

[4] CWA 14170, Security requirements for signature creation applications. May 2004. European Committee for Standardization.

[5] CWA 14167-1, Security requirements for trustworthy systems managing certificates for electronic signatures — Part 1: System Security Requirements. June 2003. European Committee for Standardization.

[6] CWA 14167-2, Security requirements for trustworthy systems managing certificates for electronic signatures — Part 2: cryptographic module for CSP signing operations with backup — Protection Profile - MCSO-PP. May 2004. European Committee for Standardization.

[7] CWA 15579, E-invoices and digital signatures. July 2006. European Committee for Standardization.

[8] Directive 1999/93/EC, Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999. Chapter 13 Volume 038 P. 50 – 58. http://eur-lex.europa.eu/smartapi/cgi/sga_doc?smartapi!celexapi!prod!CELEXnumdoc&numdoc=31999L0093&model=guichett.

[9] 2011/130/EU, Commission Decision 2011/130/EU of 25 February 2011 establishing minimum requirements for the cross-border processing of documents signed electronically by competent authorities under Directive 2006/123/EC of the European Parliament and of the Council on services in the internal market.

[10] FIPS PUB 186-3, Digital Signature Standard. National Institute of Standards and Technologies. June 2009. National Institute of Standards and Technology, U.S. Department of Commerce. http://csrc.nist.gov/publications/fips/fips186-3/fips_186-3.pdf

[11] FIPS PUB 180-4, Secure Hash Standards (SHS). March 2012. National Institute of Standards and Technology, U.S. Department of Commerce. http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf

[12] NIST SP 800-107 Revision1 Recommendation for applications using approved hash algorithms. Quynh Dang. August 2012. National Institute of Standards and Technology, U.S. Department of Commerce. http://csrc.nist.gov/publications/nistpubs/800-107-rev1/sp800-107-rev1.pdf.

[13] XMLDSIG-CORE. XML Signature Syntax and Processing (Second Edition). W3C Recommendation 10 June 2008. http://www.w3.org/TR/xmldsig-core/

[14] XMLENCR-CORE. XML Encryption Syntax and Processing. W3C Recommendation 10 December 2002. http://www.w3.org/TR/xmlenc-core/.

[15] XML Encryption Requirements. W3C Note 4 March 2002. http://www.w3.org/TR/xml-encryption-req

[16] XMLENCR-CORE1 XML Encryption Syntax and Processing Version 1.1. W3C Recommendation. 11 April 2013. http://www.w3.org/TR/xmlenc-core1/.

[17] Extensible Business Reporting Language (XBRL) 2.1 RECOMMENDATION - 2003-12-31. XBRL International (XII). http://www.xbrl.org/Specification/XBRL-RECOMMENDATION-2003-12-31.doc

[18] ZIP File Format Specification Version: 6.3.3, September 1, 2012, PKWARE Inc. http://www.pkware.com/documents/casestudies/APPNOTE.TXT

[19] NIST SP 800-57 part1, Recommendation for Key Management – Part 1: General (Revision 3). Authors: Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid. July 2012. National Institute of Standards and Technology, U.S. Department of Commerce. http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57_part1_rev3_general.pdf.

[20] NIST SP 800 131A, Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths. Authors: Elaine Barker and Allen Roginsky. January 2011. National Institute of Standards and Technology, U.S. Department of Commerce. http://csrc.nist.gov/publications/nistpubs/800-131A/sp800-131A.pdf.

[21] NIST SP 800-56B, Recommendation for Pair-Wise, Key Establishment Schemes Using Integer Factorization Cryptography. Authors: Elaine Barker, Lily Chen, Andrew Regenscheid, and Miles Smid. August 2009. National Institute of Standards and Technology, U.S. Department of Commerce. http://csrc.nist.gov/publications/nistpubs/800-56B/sp800-56B.pdf.

[22] RFC 3447, Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1. February 2003. http://www.ietf.org/rfc/rfc3447.txt.

[23] NIST SP 800-38A, Recommendation for Block Cipher Modes of Operation. Methods and Techniques. Morris Dworkin. 2001. National Institute of Standards and Technology, U.S. Department of Commerce. http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf.

[24] RegOrg Registered Organization Vocabulary. W3C Working Group Note 01 August 2013. http://www.w3.org/TR/vocab-regorg/. This is a continuative work of the EC's ISA Core Vocabularies, May 2012. https://joinup.ec.europa.eu/asset/core_business/release/100

[25] BestPractices, Best Practices on Common European Reporting Structures. Eurofiling 2013. http://www.wikixbrl.info/index.php?title=Best_Practices_on_Common_European_Reporting_Structures, http://www.xbrlwiki.info/images/c/c0/Eurofiling_header_questionnaire-results.xls.

[26] CEN, the European Committee for Standardization (CEN), Business Plan of the CEN Workshop on XBRL, Improving transparency in financial and business reporting, 30 May 2012, http://cen.eurofiling.info/wp-content/upLoads/data/BusinessPlanCENWorkshoponXBRL20120530.pdf.

[27] NIST SP 800-38D Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. Morris Dworkin. 2001. National Institute of Standards and Technology, U.S. Department of Commerce. http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf

[28] RFC 3447. Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography. Specifications Version 2.1. J. Jonsson and B. Kaliski. RSA Laboratories. The Internet Society. http://www.ietf.org/rfc/rfc3447.txt

[29] Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures

[30] Commission Decision 2011/130/EU of 25 February 2011 establishing minimum requirements for the cross-border processing of documents signed electronically by competent authorities under Directive 2006/123/EC of the European Parliament and of the Council on services in the internal market

[31] Federal Information Processing Standards Publication 186-3: Digital Signature Standard (National Institute of Standards and Technologies, U.S. Department of Commerce)

[32] Federal Information Processing Standards Publication 180-4: Secure Hash Standards (National Institute of Standards and Technologies, U.S. Department of Commerce)

[33] National Institute of Standards and Technologies, Special Publication 800-107, Recommendation for applications using approved hash algorithms

[34] W3C Recommendation XML Signature Syntax and Processing

[35] W3C Recommendation XML Encryption Syntax and Processing

[36] XBRL International (XII), Extensible Business Reporting Language (XBRL) 2.1, Recommendation – 2003-12-31

[37] PKWARE Inc., APPnotE.TXT - .ZIP File Format Specification