SLOVENSKI STANDARD
SIST-TP CEN/TR 17419-2:2023

01-april-2023

Digitalna izmenjava informacij v zavarovalniški dejavnosti - Prenos elektronskih dokumentov - 2. del: Izvajanje EN 17419-1 v odprti specifikaciji API 3.0

Digital information interchange in the insurance industry - Transfer of electronic documents - Part 2: Implementation of EN 17419-1 in Open API 3.0 specification

Digitaler Informationsaustausch in der Versicherungswirtschaft - Übertragung elektronischer Dokumente - Teil 2: Implementierung der EN 17419-1 in Open API 3.0 Spezifikation

Échange d'informations numériques dans le secteur de l'assurance - Transfert de documents électroniques - Partie 2 : Mise en uvre de l'EN 17419-1 dans la spécification Open API 3.0

Ta slovenski standard je istoveten z: **CEN/TR 17419-2:2023**

ICS:

03.060	Finance. Bančništvo. Monetarni sistemi. Zavarovanje	Finances. Banking. Monetary systems. Insurance
35.240.20	Uporabniške rešitve IT pri pisarniškem delu	IT applications in office work

SIST-TP CEN/TR 17419-2:2023 **en,fr,de**

TECHNICAL REPORT

RAPPORT TECHNIQUE

TECHNISCHER REPORT

CEN/TR 17419-2

February 2023

ICS 03.060; 35.240.20

Supersedes CEN/TR 17419-2:2021

English Version

Digital information interchange in the insurance industry - Transfer of electronic documents - Part 2: Implementation of EN 17419-1 in Open API 3.0 specification

Échange d'informations numériques dans le secteur de l'assurance - Transfert de documents électroniques - Partie 2 : Mise en ¿uvre de l'EN 17419-1 dans la spécification Open API 3.0

Digitaler Informationsaustausch in der Versicherungswirtschaft - Übertragung elektronischer Dokumente - Teil 2: Implementierung der EN 17419-1 in Open API 3.0 Spezifikation

This Technical Report was approved by CEN on 23 January 2023. It has been drawn up by the Technical Committee CEN/TC 445.

CEN members are the national standards bodies of Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Türkiye and United Kingdom.

EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

CEN-CENELEC Management Centre: Rue de la Science 23, B-1040 Brussels

Ref. No. CEN/TR 17419-2:2023 E

Contents

Page

European foreword

This document (CEN/TR 17419-2:2023) has been prepared by Technical Committee CEN/TC 445 “Digital Information Interchange in the Insurance Industry”, the secretariat of which is held by DIN.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CEN shall not be held responsible for identifying any or all such patent rights.

Any feedback and questions on this document should be directed to the users’ national standards body. A complete listing of these bodies can be found on the CEN website.

This document supersedes CEN/TR 17419-2:2021.

According to the CEN/CENELEC Internal Regulations, the national standards organisations of the following countries are bound to announce this Technical Specification: Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Türkiye and the United Kingdom.

Introduction

The EN 17419-1:2020, *Digital Information Interchange in the Insurance Industry — Transfer of electronic documents — Part 1: Process and Data Model*, defines the transfer of electronic documents between stakeholders in the insurance industry (for example between insurers and intermediaries):

- the semantic process for the transfer of documents that may be transferred as an attached file; and
- a limited number of meta data describing the document.

The definitions are described in the standard on a semantic level with process and data models in a syntax-neutral format independent from its representation in a concrete implementation syntax.

This document exemplifies a concrete implementation of the EN 17419-1:2020 as an OpenAPI specification. The OpenAPI syntax is published by the OpenAPI Initiative, an open-source collaboration project of the Linux Foundation, and is a specification for machine-readable interface files for describing, producing, consuming, and visualizing RESTful web services.

This document is a guide for organizations that want to implement the EN 17419-1:2020. Even more, the specification contained in this document can be directly implemented with OpenAPI tools that can automatically generate code, documentation and test cases.

All stakeholders that want to implement EN 17419-1:2020 will benefit from the implementation guide described in this document due to:

- Uniform implementation of EN 17419-1:2020 across the industry, based on a common technology.
- Avoidance of divergent implementations, thus avoiding incompatible digital interfaces between the stakeholders.
- Implementation for RESTful web services, a common micro-service technology.
- Specification in OpenAPI syntax, a common basis for the definition of RESTful web services.
- Automatic generation of code, documentation and test cases, based on OpenAPI tools.
- Facilitated implementation will accelerated the application of EN 17419-1:2020.
- Facilitated implementation will accelerated the usage of EN 17419 by SMEs.

1 Scope

This document specifies a concrete REST webservice API description of the processes and data (see EN 17419-1:2020 for more information) as an OpenAPI definition specified by the OpenAPI specification.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

EN 17419-1:2020, *Digital Information Interchange in the Insurance Industry - Transfer of electronic documents - Part 1: Process and Data Model*

3 Terms, definitions and abbreviations

3.1 Terms and Definitions

For the purposes of this document, the terms and definitions given in EN 17419-1:2020 apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at https://www.electropedia.org/
- ISO Online browsing platform: available at https://www.iso.org/obp

3.2 Abbreviations

API	Application Programming Interfaces
JSON	JavaScript Object Notation
OAS	OpenAPI Specification
OAI	OpenAPI Initiative
REST	Representational State Transfer
SOAP	Simple Object Access Protocol
XML	Extensible Markup Language
YAML	Yet Another Markup Language YAML Ain’t Markup Language

4 Technical basis for OpenAPI definition

4.1 Cloud services and REST

In a more and more communication based and service orientated IT infrastructure, the ease of use, implementation, operation and maintenance of IT-services as main economic success factors determine the type of underlying architectures and tools to be used. Cloud enabling of services as one strategic aspect allows to reduce the time to market of products while focussing on core competence – the business aspects - of IT activities.

REST APIs as a software architecture describe interfaces to communicate through the HTTP(S) protocol between distributed client and server systems as an alternative to SOAP. RESTful implemented services are stateless. That means, all data required for creating a service response is transmitted in the corresponding service request. Advantages of RESTful services are a better visibility, reliability and scalability.

4.2 JSON data format

IETF RFC 8259, The JavaScript Object Notation (JSON) Data Interchange Format (https://tools.ietf.org/html/rfc8259)

ECMA-262, ECMAScript® Language Specification (https://www.ecma-international.org)

With REST-Services usually the JSON data format, as a derivation or concretization of the YAML format is used, which is very slim and better human readable in contrast to the structure description orientated language of XML. JSON is more a syntactical convention for describing data in a given context by allowing an easy machine processing and parsing because of its strict structure definition.

Based on the JavaScript Programming Language Standard ECMA (http://www.ecma-international.org), the data interchange format JSON was first specified by Douglas Crockford in 1999.

4.3 YAML data format

Oren Ben-Kiki – Clark Evans – Ingy döt Net, YAML Ain't Markup Language (YAML™) Version 1.2 (https://yaml.org/spec/1.2/spec.html)

YAML is a simplified markup language to describe data in a more human readable way. It was invented in about 2001 and the most current version 1.2 (3rd Edition) was published in 2009.

YAML can be viewed as an easy to implement and use natural and consistent superset of JSON. Every JSON file is also a valid YAML file. The main focus on YAML is to provide a programming language independent data interchange format as a technical markup language with a maximum in human readability and information providing.

It consists mostly of a kind of key-value mechanism where values also may be lists (arrays) itself.

4.4 OpenAPI Specification

The OpenAPI Specification (OAS, https://www.openapis.org) defines a vendor neutral and human readable interface description for REST APIs, originally based on the Swagger Specification (Open Source Framework Swagger for HTML-Webservices) and was provided in 2016 by the OpenAPI Initiative (https://www.openapis.org). Further development and maintainance is done by the initiative and is supported of the Linux Foundation.

OpenAPI descriptions of REST APIs are implemented in OpenAPI documents. An OpenAPI document that conforms to the OpenAPI Specification is itself a JSON object, which may be represented either in JSON or YAML format.

There is an uncountable number of tools available in the world wide web for creating or dealing with OpenAPI documents. One important Open Source tool is the Swagger-Editor (https://swagger.io/tools/swagger-editor/), which allows the creation of OpenAPI conformed REST API definitions and the generation of documentation and code stubs for several client and server programming languages and runtime environments.

5 OpenAPI specification for EN 17419-1:2020

5.1 Introduction

This document describes a sample REST interface of the processes specified in EN 17419-1:2020 for Transfer of electronic documents.

The EN 17419-1:2020 itself defines the processes and the structure (data model) of the transfer of electronic documents and facilitates the transfer of electronic documents between stakeholders in the insurance industry.

This API description concentrates on a synchronous transmission process (through http method POST). Therefore a successful or unsuccessful transmission on a technical level is expressed as a direct (synchronous) response of the webservice request.

5.2 OpenAPI document

Clause 5.3 contains the complete API description of a sample REST service as an OpenAPI document in YAML format.

NOTE 1 As the OpenAPI specification defines strict rules for syntax and grammar of OpenAPI documents, so remember to especially keep the indendation of all given YAML and JSON as is to not destroy any consistency and/or semantics.

NOTE 2 Due to compatibility issues to the UN/CEFACT Core Components Library (UN/CCL) on which the data model is based on, in some classes there are several attributes defined but are not allowed to be used in explicit context situations. To take this into account for the classes in question this technical report introduces the construction of base classes containing only mandatory attributes used in all context situations. The classes in situations where all attributes are used are then derived from the base classes. As an example, within the class Communication the attribute URI of type Identifier must only use the attribute Identifier.Content. Therefore IdentifierBase is introduced as base class with only one attribute Content and URI is defined as IdentifierBase instead of Identifier. The derived class Identifier extends IdentifierBase with the other attributes IdentificationScheme, IdentificationSchemeAgency, IdentificationSchemeVersion and may then be used in situations where all attributes of Identifier are used, e. g. for attribute CountryIdentifier in class Location.

5.3 OpenAPI document in YAML format

```
openapi: 3.0.3
info:
  description: |
    This specification describes a sample REST interface of the processes
specified in the European standard EN 17419-1:2020 for Transfer Of
Electronic Documents.

    The European standard (EN 17419-1:2020) itself defines the processes
and the structure (data model) of the transfer of electronic documents,
and facilitates the transfer of electronic documents between stakeholders
in the insurance industry.

    This API description implements the EN17419-1 as a synchronous
transmission process (post).
    The technical aknowledgement therefore is provided in the
transmitInsuranceTransaction response.

    Last edited on 25th, November 2020
  version: '1.1.7'
  contact:
```

```
    name: CEN TC445
    url: http://tc445.info
    email: info@tc445.info
  title: TOED - Transfer Of Electronic Documents - Technical Report
EN17419-2
servers:
  - description: 'localhost:8080'
    url: http://localhost:8080/cen-tc445/TOED/V1
paths:
  /transmitInsuranceTransaction:
    post:
      tags:
        - Insurance Transaction
      summary: |
        Transmits an Insurance Transaction object with all relevant
content (meta data and link to binary files). The sender prepares the
object InsuranceTransaction with its content and transfers this
InsuranceTransaction to the receiver.
      operationId: transmitInsuranceTransaction
      responses:
        '200':
          description: |
            successful operation. The details of the transmission are
returned in the transmission status message of the response within the
Event object.
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Event'
              examples:
                eventSuccessfullExample:
                  $ref: '#/components/examples/eventSuccessfullExample'
        '400':
          description: Invalid Insurance Transaction
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Event'
              examples:
                eventUnsuccessfullExample:
                  $ref: '#/components/examples/eventUnsuccessfullExample'
      requestBody:
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/InsuranceTransaction'
            examples:
              insuranceTransaction71Example:
                $ref:
'#/components/examples/insuranceTransaction71Example'
              insuranceTransaction72Example:
                $ref:
'#/components/examples/insuranceTransaction72Example'
              insuranceTransaction73Example:
```

```
                $ref:
'#/components/examples/insuranceTransaction73Example'
              insuranceTransaction74Example:
                $ref:
'#/components/examples/insuranceTransaction74Example'
              insuranceTransaction75Example:
                $ref:
'#/components/examples/insuranceTransaction75Example'
              insuranceTransaction76Example:
                $ref:
'#/components/examples/insuranceTransaction76Example'
              insuranceTransaction77Example:
                $ref:
'#/components/examples/insuranceTransaction77Example'
              insuranceTransaction78Example:
                $ref:
'#/components/examples/insuranceTransaction78Example'
        description: Insurance Transaction to transmit to the receiver
        required: true

components:
  schemas:

    CodeBase:
      description: |
        Information used to identify and distinguish uniquely one
instance of an object in a code list from all other objects within the
same code list.
        Base Code object where only attribut Content is needed.
      type: object
      required:
        - Content
      properties:
        Content:
          description: 'The unique character string identifying the
code.'
          type: string

    Code:
      description: 'Information used to identify and distinguish uniquely
one instance of an object in a code list from all other objects within
the same code list.'
      allOf:
      - $ref: '#/components/schemas/CodeBase'
      - type: object
        properties:
          CodeList:
            description: 'The identification of a list of codes.'
            type: string
          CodeListAgency:
            description: 'The identification of the agency that maintains
the code list. The identification shall be an entry of UN/CEFACT code
list 3055.'
            type: string
          CodeListVersion:
```

```
            description: 'The version of the code list.'
            type: string

    IdentifierBase:
      description: |
        'Information used to identify and distinguish uniquely one
instance of an object in an identification scheme from all other objects
within the same scheme.
        Base Identifier object where only attribut Content is needed.'
      type: object
      required:
        - Content
      properties:
        Content:
          description: 'The character string of the identifier.'
          type: string

    Identifier:
      description: 'Information used to identify and distinguish uniquely
one instance of an object in an identification scheme from all other
objects within the same scheme.'
      allOf:
      - $ref: '#/components/schemas/IdentifierBase'
      - type: object
        properties:
          IdentificationScheme:
            description: 'The identification of the identification
scheme.'
            type: string
          IdentificationSchemeAgency:
            description: 'The identification of the agency that maintains
the identification scheme. The identification shall be an entry of
UN/CEFACT code list 3055.'
            type: string
          IdentificationSchemeVersion:
            description: 'The version of the identification scheme.'
            type: string

    Location:
      description: 'A physical location or place.'
      type: object
      properties:
        Name:
          description: 'A name, expressed as text, of this location.'
          type: array
          items:
            type: string
        CountryIdentifier:
          description: 'A unique identifier of a country for this
location. The value in CountryIdentifier.Content shall be an entry of the
code list ISO 3166 Alpha-2. In CountryIdentifier.IdentificationScheme
"3166-Alpha-2" shall be specified. In
CountryIdentifier.IdentificationSchemeAgency "5" (code entry for "ISO" in
the UN/CEFACT code list 3055) shall be specified.'
          type: array
```

```
          items:
            allOf:
            - $ref: '#/components/schemas/Identifier'

    BinaryFile:
      description: 'Digital representation of a document.'
      type: object
      properties:
        FileName:
          description: 'The file name, expressed as text, of this binary
file.'
          type: string
        URI:
          description: 'A unique Uniform Resource Identifier (URI) for
this binary file. This identifier shall be specified in URI.Content. The
other attributes in URI shall not be used.'
          type: array
          items:
            allOf:
            - $ref: '#/components/schemas/IdentifierBase'
        Encoding:
          description: 'A code specifying the encoding of this binary
file. The value in Encoding.Content shall be an entry of the code list
ISO IEC 10646 or ISO IEC 8859-15. In Encoding.CodeList "10646" or "8859-
15" shall be specified. In Encoding.CodeListAgency "5" (code entry for
"ISO" in the UN/CEFACT code list 3055) shall be specified.'
          type: array
          items:
            allOf:
            - $ref: '#/components/schemas/Code'
        Description:
          description: 'A textual description of this binary file.'
          type: string

    Contract:
      description: 'An agreement between two or more parties, especially
one that is written or spoken and enforceable by law.'
      type: object
      properties:
        MainBusinessClass:
          description: |
            'The code specifying the main class of business for this
contract. The value in MainBusinessClass.Content shall be an entry of the
code list specified in AnnexA.3. In MainBusinessClass.CodeList
"EN17419:2020A3" shall be specified. In MainBusinessClass.CodeListAgency
"403" (code entry for "CEN" in the UN/CEFACT code list 3055) shall be
specified. Additionally one or more market specific codes may be
specified. If used, for each code the code entry shall be given in
MainBusinessClass.Content. In MainBusinessClass.CodeList the
identification of the market specific code list shall be specified. The
agency responsible for this code list shall be specified in
MainBusinessClass.CodeListAgency with a code entry from the UN/CEFACT
code list 3055.'
          type: array
          items:
```

```
            allOf:
            - $ref: '#/components/schemas/Code'
        SecondaryBusinessClass:
          description: |
            A code specifying a secondary class of business for this
contract.
            One or more market specific codes may be specified. For each
code the code entry shall be given in SecondaryBusinessClass.Content. In
SecondaryBusinessClass.CodeList the identification of the market specific
code list shall be specified. The agency responsible for this code list
shall be specified in SecondaryBusinessClass.CodeListAgency with a code
entry from the UN/CEFACT code list 3055.
          type: array
          items:
            allOf:
            - $ref: '#/components/schemas/Code'
        ProvidedIdentity:
          description: |
            'An identification provided for this contract. The policy
number as given by the party issuing this number e.g. the insurance
company or the insurance intermediary.'
          type: array
          items:
            allOf:
            - $ref: '#/components/schemas/Identity'

    Communication:
      description: |
        'The exchange of thoughts, messages, or information, as by
speech, signals, writing, or behaviour between persons and/or
organizations.'
      type: object
      properties:
        URI:
          description: |
            The unique identifier of the Uniform Resource Identifier
(URI) for this communication, such as a web or an email address.
            This identifier shall be specified in URI.Content. The other
attributes in URI shall not be used.
          allOf:
          - $ref: '#/components/schemas/IdentifierBase'
        Channel:
          description: |
            'The code specifying the channel or manner in which a
communication can be made, such as telephone or email. The value in
Channel.Content shall be an entry of the code list UN/CEFACT 3155. In
Channel.CodeList "3155" shall be specified. In Channel.CodeListAgency "6"
(code entry for "UN/ECE" in the UN/CEFACT code list 3055) shall be
specified.'
          allOf:
          - $ref: '#/components/schemas/Code'
        CompleteNumber:
          description: |
            'A text string of characters that make up the complete number
for this communication.'
```

```
          type: string

    Person:
      description: 'An individual human being.'
      type: object
      properties:
        GivenName:
          description: |
            'Name or names, expressed as text, usually given to a person
by his/her parents at birth.'
          type: array
          items:
            type: string

    Address:
      description: 'The location at which a particular organization or
person may be found or reached.'
      type: object
      properties:
        Postcode:
          description: |
            'A code specifying the postcode of the address. This code
shall be specified in Postcode.Content. The other attributes in Postcode
shall not be used.'
          type: array
          items:
            allOf:
            - $ref: '#/components/schemas/CodeBase'
        PostOfficeBox:
          description: |
            'The unique identifier, expressed as text, of a container
commonly referred to as a box, in a post office or other postal service
location, assigned to a person or organization, where postal items may be
kept for this address.'
          type: string
        BuildingNumber:
          description: 'The number, expressed as text, of a building or
house on a street at this address.'
          type: string
        RoomIdentification:
          description: 'The identification, expressed as text, of a room,
suite, office or apartment as part of an address.'
          type: string
        StreetName:
          description: 'A name, expressed as text, of a street or
thoroughfare.'
          type: array
          items:
            type: string
        CityName:
          description: 'The name, expressed as text, of the city, town or
village of this address.'
          type: string
        AttentionOf:
          description: |
```