

E-Rechnung: Technische Details beim Empfangen, Validieren, Erstellen und Versenden

Dr. Philipp Liegl

E-Mail: philipp.liegl@ecosio.com

<https://ecosio.com>





About me



Philipp Liegl

Wirtschaftsinformatiker mit Schwerpunkt B2B-Prozesse

Mitarbeit in der UN/CEFACT-Standardisierung und der e-Rechnungs-Standardisierung

Über 20 Jahre Projekterfahrung in den Bereichen FMCG, Einzelhandel, Fertigung, Industrie und Automotive

Technischer Projektleiter bei verschiedenen internationalen e-Rechnungsprojekten

Tüftelt immer noch mit Begeisterung an Integrationslösungen - vor allem mit SAP



Agenda

Rechnungen empfangen

Rechnungen validieren

Rechnungen versenden

Zielarchitektur

Rechnungen empfangen



Eine elektronische Rechnung muss im XML-Format gemäß EN 16931 vorliegen

Extensible Markup Language

Unterscheidung zwischen Markup (Struktur) und dem eigentlichen Inhalt

- Markup: Rot und Orange
- Inhalt: Weiß und Grün

Verwendete Konzepte

- Definition der Struktur: **XML Schema**
- Integritätsbedingungen: **Schematron**
- Selektion von Elementen/Attributen: **XPath**
- Einzelnes Dokument nennt man auch **XML-Instanz**

```
<?xml version="1.0" encoding="UTF-8"?>
<Invoice xmlns:cac="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2"
xmlns:cbc="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2"
xmlns:qdt="urn:oasis:names:specification:ubl:schema:xsd:QualifiedDataTypes-2"
xmlns:udt="urn:oasis:names:specification:ubl:schema:xsd:UnqualifiedDataTypes-2"
xmlns:ccts="urn:un:unece:uncefact:documentation:2"
xmlns="urn:oasis:names:specification:ubl:schema:xsd:Invoice-2">
  <cbc:CustomizationID>urn:cen.eu:en16931:2017#compliant#urn:xoev-de:kosit:standard:xrechnung_1.2</
  cbc:CustomizationID>
  <cbc:ID>12115118</cbc:ID>
  <cbc:IssueDate>2015-01-09</cbc:IssueDate>
  <cbc:DueDate>2015-01-09</cbc:DueDate>
  <cbc:InvoiceTypeCode>380</cbc:InvoiceTypeCode>
  <cbc:DocumentCurrencyCode>EUR</cbc:DocumentCurrencyCode>
  <cbc:BuyerReference>A test buyer reference</cbc:BuyerReference>
  <cac:AccountingSupplierParty>
    <cac:Party>
      <cac:PostalAddress>
        <cbc:StreetName>Postbus 71</cbc:StreetName>
        <cbc:CityName>Velsen-Noord</cbc:CityName>
        <cbc:PostalZone>1950 AB</cbc:PostalZone>
        <cac:Country>
          <cbc:IdentificationCode>NL</cbc:IdentificationCode>
        </cac:Country>
      </cac:PostalAddress>
      <cac:PartyTaxScheme>
        <cbc:CompanyID>NL8200.98.395.B.01</cbc:CompanyID>
        <cac:TaxScheme>
          <cbc:ID>VAT</cbc:ID>
        </cac:TaxScheme>
      </cac:PartyTaxScheme>
      <cac:PartyLegalEntity>
        <cbc:RegistrationName>De Koksmaat</cbc:RegistrationName>
        <cbc:CompanyID>57151520</cbc:CompanyID>
      </cac:PartyLegalEntity>
    </cac:Party>
  </cac:AccountingSupplierParty>
  <cac:Contact>
```



EU-Richtlinie 2014/55/EU und die EN 16931

Schreibt die **Entwicklung eines einheitlichen EU-weiten E-Rechnungsstandards** vor

Verpflichtet den Einsatz der elektronischen Rechnung im Bereich der öffentlichen Beschaffung in Europa

Öffentliche Auftraggeber müssen elektronische Rechnungen von Lieferanten akzeptieren

Die **EU-Richtlinie resultierte in der EN 16931** (syntax-neutral)

Zwei offiziell unterstützte Syntaxen

- Universal Business Language (UBL)
- Cross Industry Invoice (CII)

Official Journal

of the European Union



English edition

Legislation

ISSN 1977-0677

L 133

Volume 57
6 May 2014

Contents

I Legislative acts

page

DIRECTIVES

* **Directive 2014/55/EU of the European Parliament and of the Council of 16 April 2014 on electronic invoicing in public procurement (¹)** 1



Einsatz der EN 16931: XRechnung in Deutschland

<https://xeinkauf.de/xrechnung/>

Die XRechnung ist eine Core Invoice Usage Specification (CIUS) der Europäischen Norm EN 16931

Sie enthält Einschränkungen/Erweiterungen der EN 16931 speziell für Deutschland (Verwendung von Leitweg-ID, Abbildung von Skonto, Subline-Items, etc.)

Aktuelle Version: 3.0.1 (gültig seit 01.02.2024)

Gültige Syntaxen:

- **Cross Industry Invoice (CII)**
 - XML-Standard, der bei ZUGFeRD/Factur-X verwendet wird
 - Eine ZUGFeRD 2.2 Rechnung ist auch eine gültige XRechnung (nur wenn das Profil EN 16931 verwendet wird)
- **Universal Business Language (UBL)**
 - XML-basierter Standard für elektronische Dokumente
 - Wird bei Peppol standardmäßig verwendet
- **ZUGFeRD/Factur-X**
 - PDF/A-3 mit eingebettetem CII XML



Beispiel für EN16931/XRechnung

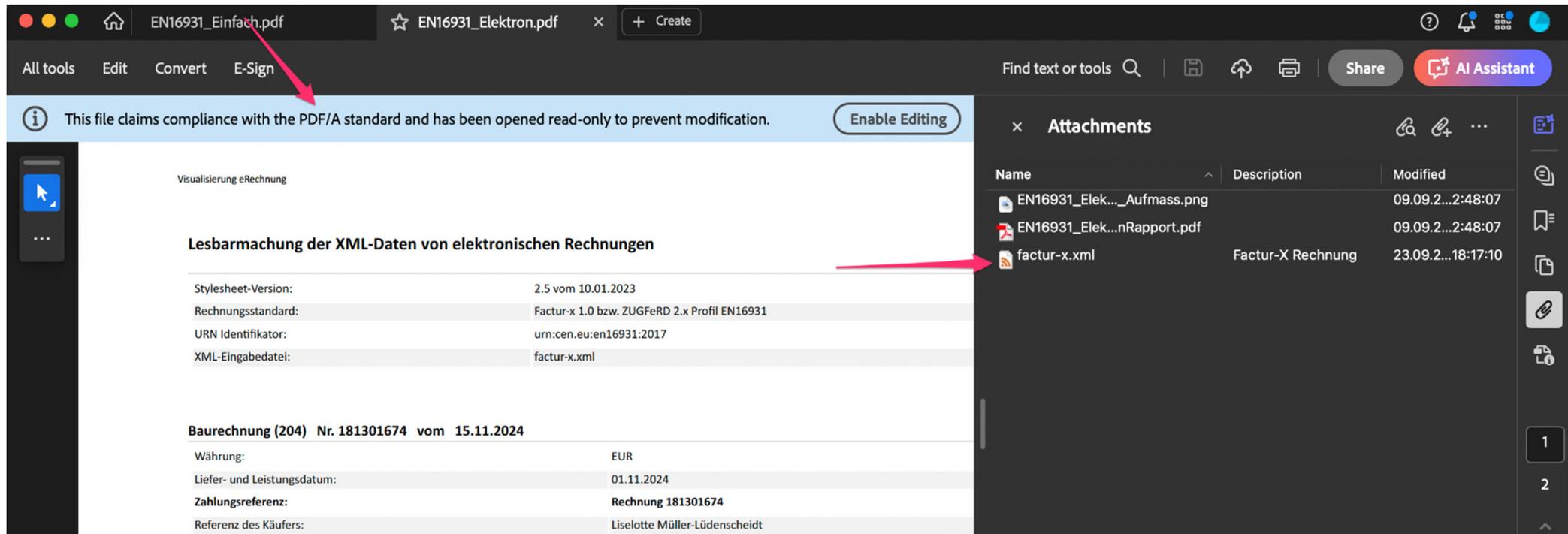
UBL - Universal Business Language

```
<?xml version="1.0" encoding="UTF-8"?>
<ubl:Invoice xmlns:ubl="urn:oasis:names:specification:ubl:schema:xsd:Invoice-2"
  xmlns:cac="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2"
  xmlns:cbc="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:oasis:names:specification:ubl:schema:xsd:Invoice-2 http://docs.oasis-open.org/ubl/os-
  <cbc:CustomizationID>urn:cen.eu:en16931:2017#compliant#urn:xoev-de:kosit:standard:xrechnung_1.2</cbc:CustomizationID>
  <cbc:ID>PRG1502112</cbc:ID>
  <cbc:IssueDate>2015-04-24+01:00</cbc:IssueDate>
  <cbc:InvoiceTypeCode>380</cbc:InvoiceTypeCode>
  <cbc:Note>#ADU#Trainer: Herr [...]</cbc:Note>
  <cbc:DocumentCurrencyCode>EUR</cbc:DocumentCurrencyCode>
  <cbc:TaxCurrencyCode>EUR</cbc:TaxCurrencyCode>
  <cbc:BuyerReference>99000000-18188-16</cbc:BuyerReference>
  <cac:InvoicePeriod>
    <cbc:StartDate>2015-04-20+01:00</cbc:StartDate>
    <cbc:EndDate>2015-04-24+01:00</cbc:EndDate>
  </cac:InvoicePeriod>
  <cac:ProjectReference>
    <cbc:ID>PR456789</cbc:ID>
  </cac:ProjectReference>
```

CII - Cross Industry Invoice

```
<?xml version="1.0" encoding="UTF-8"?>
<rsm:CrossIndustryInvoice xmlns:rsm="urn:un:unece:uncefact:data:standard:CrossIndustryInvoice:100"
  xmlns:ram="urn:un:unece:uncefact:data:standard:ReusableAggregateBusinessInformationEntity:100"
  xmlns:udt="urn:un:unece:uncefact:data:standard:UnqualifiedDataType:100"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:un:unece:uncefact:data:standard:CrossIndustryInvoice:100
  <rsm:ExchangedDocumentContext>
  <ram:GuidelineSpecifiedDocumentContextParameter>
    <ram:ID>urn:cen.eu:en16931:2017#compliant#urn:xoev-de:kosit:standard:xrechnung_1.2</ram:ID>
  </ram:GuidelineSpecifiedDocumentContextParameter>
</rsm:ExchangedDocumentContext>
<rsm:ExchangedDocument>
  <ram:ID>PRG1502112</ram:ID>
  <ram:TypeCode>380</ram:TypeCode>
  <ram:IssueDateTime>
    <udt:DateTimeString format="102">20150424</udt:DateTimeString>
  </ram:IssueDateTime>
  <ram:IncludedNote>
    <ram:Content>Trainer: Herr [...]</ram:Content>
    <ram:SubjectCode>ADU</ram:SubjectCode>
  </ram:IncludedNote>
```

ZUGFeRD/Factor-X



The screenshot shows a PDF viewer interface. The main content area displays the 'Visualisierung eRechnung' (e-invoice visualization) for a Factor-X invoice. The invoice details are as follows:

Lesbarmachung der XML-Daten von elektronischen Rechnungen	
Stylesheet-Version:	2.5 vom 10.01.2023
Rechnungsstandard:	Factor-x 1.0 bzw. ZUGFeRD 2.x Profil EN16931
URN Identifikator:	urn:cen.eu:en16931:2017
XML-Eingabedatei:	factur-x.xml

Baurechnung (204) Nr. 181301674 vom 15.11.2024	
Währung:	EUR
Liefer- und Leistungsdatum:	01.11.2024
Zahlungsreferenz:	Rechnung 181301674
Referenz des Käufers:	Liselotte Müller-Lüdenscheidt

The right-hand side of the viewer shows an 'Attachments' panel with the following table:

Name	Description	Modified
EN16931_Elek..._Aufmass.png		09.09.2...2:48:07
EN16931_Elek...nRapport.pdf		09.09.2...2:48:07
factur-x.xml	Factor-X Rechnung	23.09.2...18:17:10

Red arrows in the image point to the 'E-Sign' menu item in the top toolbar and the 'factur-x.xml' attachment in the right panel.

- Das PDF muss dem PDF/A-3-Standard genügen
- Es muss genau ein Attachment mit dem Namen **factur-x.xml** oder **xrechnung.xml** in der PDF/A-3-Datei vorhanden sein
- Weitere Attachments (z.B. rechnungsbegleitende Unterlagen) sind zulässig
- XMP-Metadaten gemäß ZUGFeRD-Standard müssen eingebettet sein



ZUGFeRD/Factor-X



EN16931_Einfach.pdf

All tools Edit Convert E-Sign

This file claims compliance with the PDF/A standard and h

Visualisierung eRechnung

Lesbarmachung der XML-Daten von

Stylesheet-Version:
Rechnungsstandard:
URN Identifikator:
XML-Eingabedatei:

Baurechnung (204) Nr. 181301674 vom 15.11.2024

Währung:	EUR
Liefer- und Leistungsdatum:	01.11.2024
Zahlungsreferenz:	Rechnung 181301674
Referenz des Käufers:	Liselotte Müller-Lüdenscheidt
Buchungsreferenz:	420
Weitere Referenz:	13130162
	Art der Referenz:Referenzpapier (916)
Weitere Referenz:	42389
	Art der Referenz:Referenzpapier (916)

```

:rsm:CrossIndustryInvoice xmlns:rsm="urn:un:unece:uncefact:data:standard:CrossIndustryInvoice:100"
  <rsm:SupplyChainTradeTransaction>
    <ram:IncludedSupplyChainTradeLineItem>
    </ram:IncludedSupplyChainTradeLineItem>
    <ram:ApplicableHeaderTradeAgreement>
      <ram:BuyerReference>Liselotte Müller-Lüdenscheidt</ram:BuyerReference>
      <ram:SellerTradeParty>
        <ram:ID>549910</ram:ID>
        <ram:Name>ELEKTRON Industrieservice GmbH</ram:Name>
        <ram:Description>Geschäftsführer Egon Schrempf Amtsgericht Stuttgart HRB 1234</ram:
        <ram:PostalTradeAddress>
          <ram:PostcodeCode>74465</ram:PostcodeCode>
          <ram:LineOne>Erfurter Strasse 13</ram:LineOne>
          <ram:CityName>Demoort</ram:CityName>
          <ram:CountryID>DE</ram:CountryID>
        </ram:PostalTradeAddress>
        <ram:SpecifiedTaxRegistration>
          <ram:ID schemeID="VA">DE136695976</ram:ID>
        </ram:SpecifiedTaxRegistration>
      </ram:SellerTradeParty>

```

1

2

↑

↓

↻

📄

Rechnungen validieren



Wann ist eine Rechnung eine gültige Rechnung?

Validierung einer XML-Instanz

Schritt 1: Sicherstellung der Wohlgeformtheit

- z.B. *end tag* für alle *start tags*
- Siehe <https://www.w3resource.com/xml/well-formed.php>

Schritt 2: Sicherstellung der XML-Schemakonformität

- Erfüllt die XML-Instanz die Regeln des XML-Schemas?
- Entspricht das Encoding der XML-Instanz den Vorgaben des XML-Schemas? (zumeist UTF-8)

Schritt 3: Sicherstellung der Schematron-Konformität

- Entspricht die XML-Instanz den Schematron-Regeln der EN16931 (wenn/dann-Bedingungen)
- Siehe <https://github.com/ConnectingEurope/eInvoicing-EN16931/tree/master/ubl/schematron>

Beispiel für eine XML-Schemavalidierung

```
1 package com.ecosio.ihkdemo;
2
3
4 import java.io.File;
5 import java.io.IOException;
6 import javax.xml.XMLConstants;
7 import javax.xml.transform.stream.StreamSource;
8 import javax.xml.validation.Schema;
9 import javax.xml.validation.SchemaFactory;
10 import javax.xml.validation.Validator;
11 import org.xml.sax.SAXException;
12
13 public class XMLValidation {
14
15     public static void main(String[] args) {
16         System.out.println("invoice.xml validates against EN16931.xsd? "+validateXMLSchema("Invoice-EN16931.xsd", "invoice.xml"));
17     }
18
19     public static boolean validateXMLSchema(String xsdPath, String xmlPath) {
20         try {
21             SchemaFactory factory =
22                 SchemaFactory.newInstance(XMLConstants.W3C_XML_SCHEMA_NS_URI);
23             Schema schema = factory.newSchema(new File(xsdPath));
24             Validator validator = schema.newValidator();
25             validator.validate(new StreamSource(new File(xmlPath)));
26         } catch (IOException | SAXException e) {
27             System.out.println("Exception: "+e.getMessage());
28             return false;
29         }
30         return true;
31     }
32 }
```



Schematron-Regeln der EN16931

```
<rule context="/ubl:Invoice | /cn:CreditNote">
  <assert id="BR-01" flag="fatal" test="normalize-space(cbc:CustomizationID) != ''">[BR-01]-An Invoice shall have a Specification identifier (BT-24). </assert>
  <assert id="BR-02" flag="fatal" test="normalize-space(cbc:ID) != ''">[BR-02]-An Invoice shall have an Invoice number (BT-1).</assert>
  <assert id="BR-03" flag="fatal" test="normalize-space(cbc:IssueDate) != ''">[BR-03]-An Invoice shall have an Invoice issue date (BT-2).</assert>
  <assert id="BR-04" flag="fatal" test="normalize-space(cbc:InvoiceTypeCode) != '' or normalize-space(cbc:CreditNoteTypeCode) != ''">[BR-04]-An Invoice shall have an Invoice type code (BT-3) or a Credit Note type code (BT-4).</assert>
  <assert id="BR-05" flag="fatal" test="normalize-space(cbc:DocumentCurrencyCode) != ''">[BR-05]-An Invoice shall have an Invoice currency code (BT-5).</assert>
  <assert id="BR-06" flag="fatal" test="normalize-space(cac:AccountingSupplierParty/cac:Party/cac:PartyLegalEntity/cbc:RegistrationName) != ''">[BR-06]-An Invoice shall have a Seller registration name (BT-6).</assert>
  <assert id="BR-07" flag="fatal" test="normalize-space(cac:AccountingCustomerParty/cac:Party/cac:PartyLegalEntity/cbc:RegistrationName) != ''">[BR-07]-An Invoice shall have a Buyer registration name (BT-7).</assert>
  <assert id="BR-08" flag="fatal" test="exists(cac:AccountingSupplierParty/cac:Party/cac:PostalAddress)">[BR-08]-An Invoice shall contain the Seller postal address (BT-8).</assert>
  <assert id="BR-10" flag="fatal" test="exists(cac:AccountingCustomerParty/cac:Party/cac:PostalAddress)">[BR-10]-An Invoice shall contain the Buyer postal address (BT-9).</assert>
  <assert id="BR-16" flag="fatal" test="exists(cac:InvoiceLine) or exists(cac:CreditNoteLine)">[BR-16]-An Invoice shall have at least one Invoice line (BT-10) or one Credit Note line (BT-11).</assert>
  <assert id="BR-53" flag="fatal" test="every $taxcurrency in cbc:TaxCurrencyCode satisfies exists(//cac:TaxTotal/cbc:TaxAmount[@currencyID=$taxcurrency])">[BR-53]-An Invoice shall have a tax total (BT-12) for each tax currency (BT-13).</assert>
  <assert id="BR-66" flag="fatal" test="count(cac:PaymentMeans/cac:CardAccount) &lt;= 1">[BR-66]-An Invoice shall contain maximum one Payment Card account (BT-14).</assert>
  <assert id="BR-67" flag="fatal" test="count(cac:PaymentMeans/cac:PaymentMandate) &lt;= 1">[BR-67]-An Invoice shall contain maximum one Payment Mandate (BT-15).</assert>
```



XML-Instanz gegen Schematron validieren

The screenshot shows the GitHub repository page for 'ph-schematron'. The header features the project name 'ph-schematron' in a large, white, handwritten-style font, with the tagline 'Java Schematron library that supports XSLT and native application' below it. A 'View project on GitHub' button with the GitHub logo is in the top right. The main content area has a light blue background and includes a section header 'ph-schematron' with a slash icon. Below this is a paragraph describing the library's purpose and capabilities. A 'Prerequisites' section follows, detailing the knowledge required to use the library. On the right side, there are three blue buttons: 'Download .zip file', 'Download .tar.gz file', and 'Maven site 5.0.8'. A 'Donate' button with various payment logos is also visible.

ph-schematron

Java Schematron library that supports XSLT and native application

View project on GitHub

ph-schematron

ph-schematron is a Java library that validates XML documents via [ISO Schematron](#). It offers several different possibilities to perform this task where each solution offers its own advantages and disadvantages that are outlined below in more detail. ph-schematron only supports ISO Schematron and no other Schematron version. The most common way is to convert the source Schematron file to an XSLT script and apply this XSLT on the XML document to be validated. Alternatively ph-schematron offers a native implementation for the Schematron XPath binding which offers superior performance over the XSLT approach but has some other minor limitations.

Prerequisites

It is assumed that you have a basic knowledge what Schematron is, and what Schematron can do for you. A good introduction can be found in Dave Pawsons Schematron tutorial at <http://www.dpawson.co.uk/schematron/>. It is also assumed that you have basic knowledge of the Java language, so that you can understand the code examples, that you have at least basic understanding of XSLT (Extensible Stylesheet Language Transformations) and that you have good knowledge of XML itself.

Download .zip file

Download .tar.gz file

is maintained by Philip Helger.

Donate

VISA, Mastercard, American Express, PayPal

Maven site 5.0.8

<https://github.com/phax/ph-schematron>



Schema + Schematron Validierung als kostenfreies Online-Service

VALIDATE PEPPOL AND XML FILES

FREE PEPPOL AND XML DOCUMENT VALIDATOR

Easily validate Peppol and XML documents such as EN 16931 (e.g., XRechnung), EHF, OIOUBL, A-NZ PEPPOL BIS3, CII Cross-Industry Invoice, OpenPEPPOL formats, various UBL file types, and many more.

<https://ecosio.com/en/peppol-and-xml-document-validator/>



ZUGFeRD validieren



Registrierung Anmelden

Unterstützer Infothek Blog Events Open Community Kontakt



- USERGROUP
- FORMAT EXTENSIONS
- ZF/FX VALIDATION**
- ZF/FX INVOICE CREATION
- ZF/FX ONLINE FORUM

ZF/FX Validation

Aufgrund der gesetzlichen Anforderungen zur innerdeutschen E-Rechnungspflicht ab 01. Januar 2025 haben wir ZF/FX Validation gemeinsam mit unserer Schwestergesellschaft OXSEED sowie den Entwicklern von Mustang und XML.Valitool zu einem Validierungs- und Visualisierungstool weiterentwickelt.

No file chosen

https://www.zugferd-community.net/de/open_community/validation

Rechnungen versenden

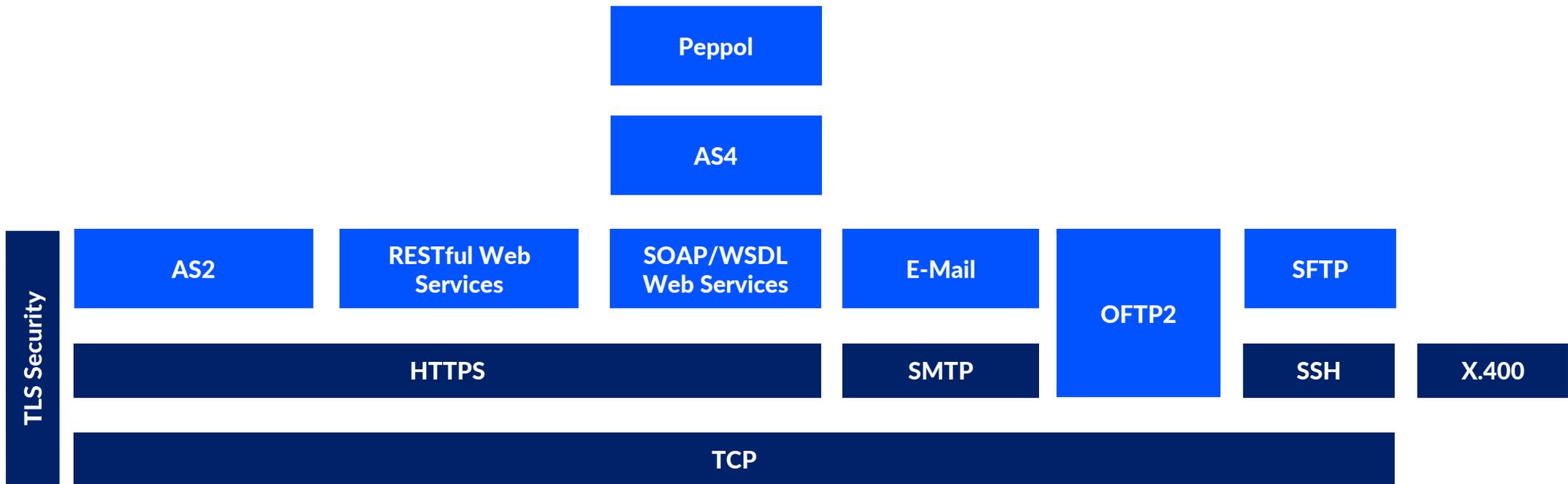


Die neue gesetzliche Regelung enthält **keine**
Vorgaben zum Übermittlungsweg von
elektronischen Rechnungen.



In Frage kommen als prinzipiell “viele” potenzielle Übertragungswege

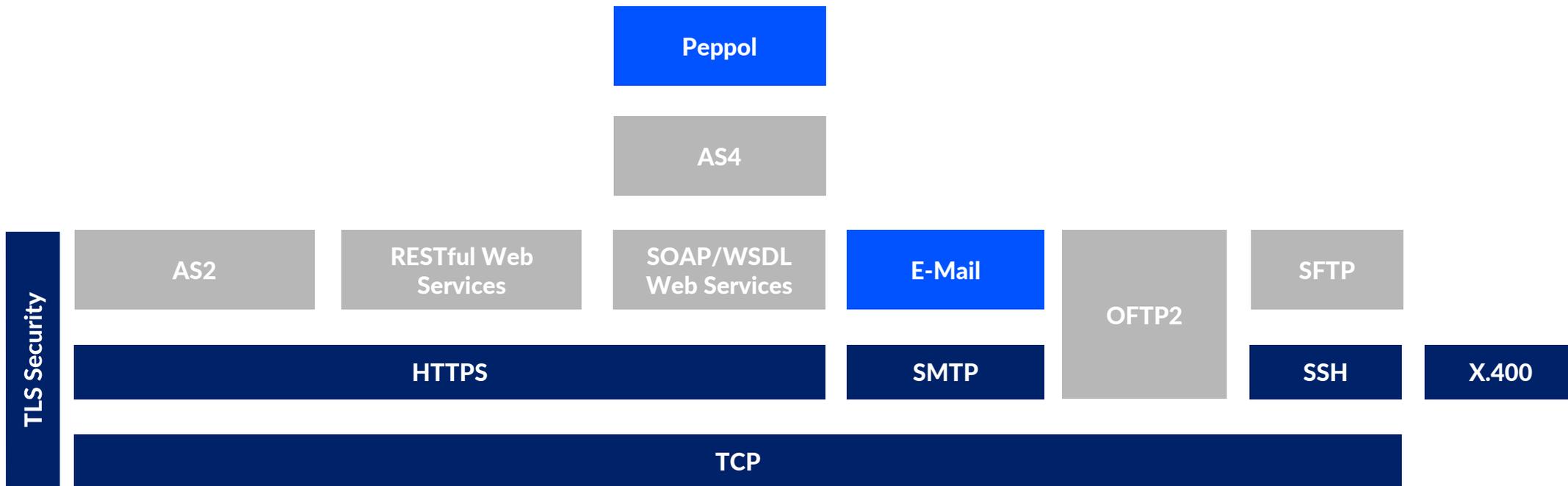
Beispiel-Protokolle aus dem B2B-Bereich





Realistisch sind aber nur jene, die keine vorherige Abstimmung erfordern

Beispiel-Protokolle aus dem B2B-Bereich





Herausforderungen bei E-Mail

- Fehlende Nichtabstreitbarkeit des Erhalts (kein Acknowledgement)
- Keine Zustellgarantie
 - Mailbox voll
 - Absender-Adresse blockiert
 - ...
- Vollkommen offen - jeder kann an jeden irgendetwas senden
- Keine Validierung von Rechnungen vor dem Versand
- Spam





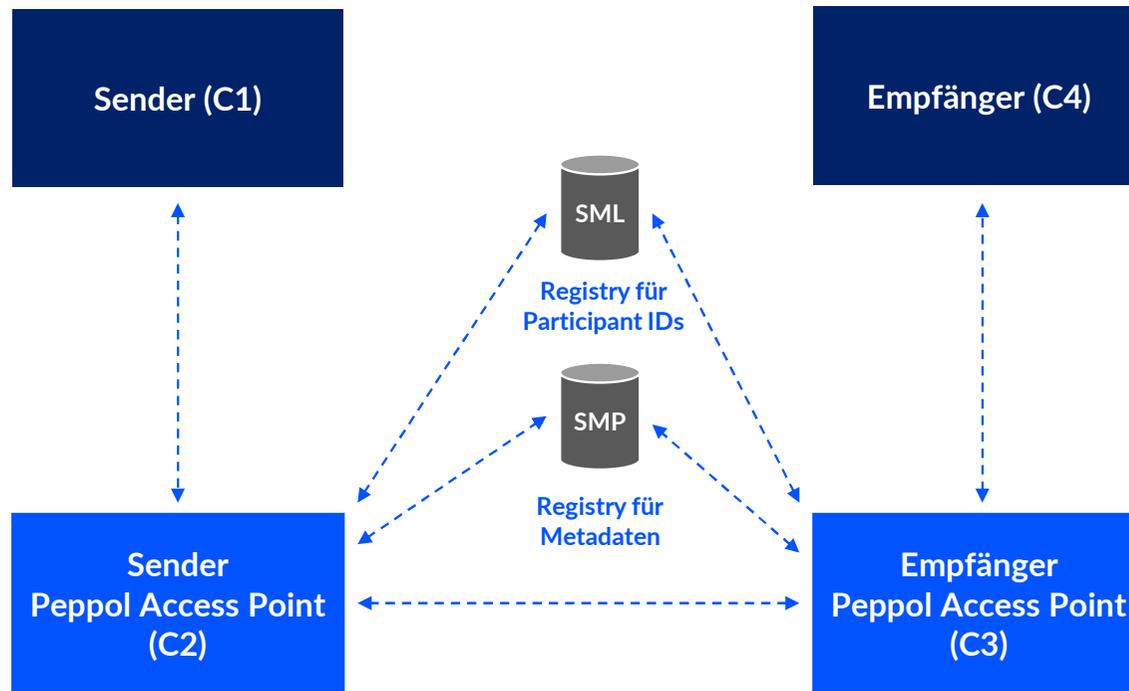
Was wäre wenn?



Source: https://commons.wikimedia.org/wiki/File:Crowd_of_people_with_phones.jpg



Peppol-Übersicht



- Four-corner Modell für Cross-Border E-Procurement
- Sender und Empfänger können einen beliebigen Access Point wählen
- “Connect once - connect to all“
- Keine Roaming-Gebühren zwischen den APs

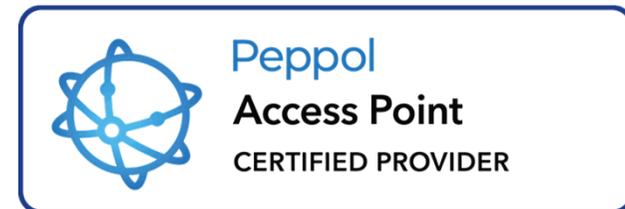


Was braucht ein Sender, um Ihnen eine Nachricht über Peppol zu senden?

Zwei Dinge:

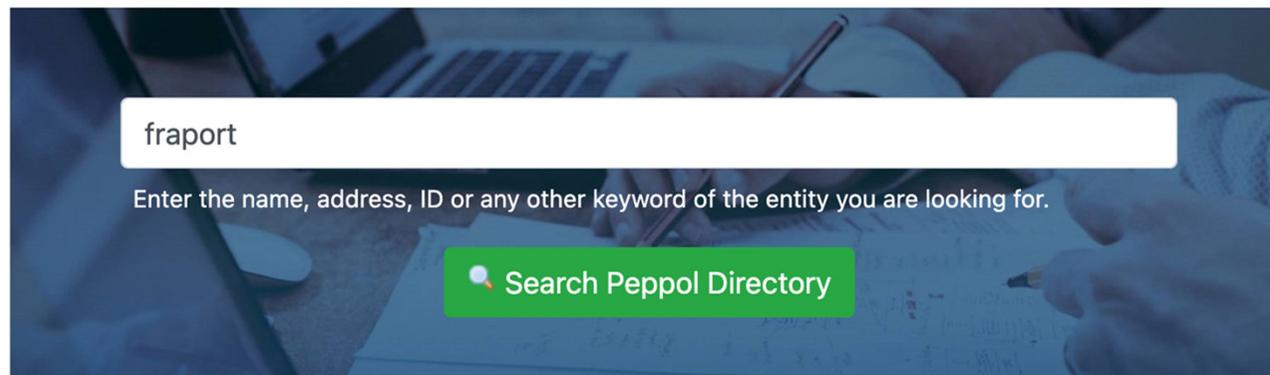
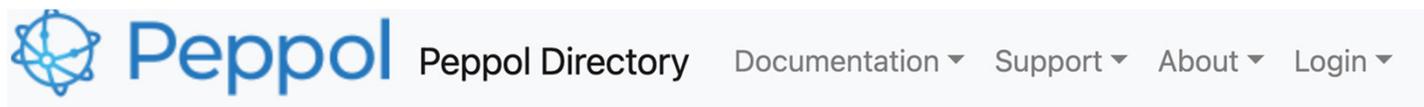
- A) Ihre Peppol ID - z.B. 9930:DE265038025

- A) Die Rechnung oder Gutschrift im UBL-Format



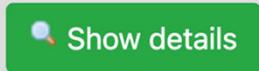


Voraussetzung für den Empfang über Peppol: registrierte Peppol ID



Found 2 entities matching 'fraport'

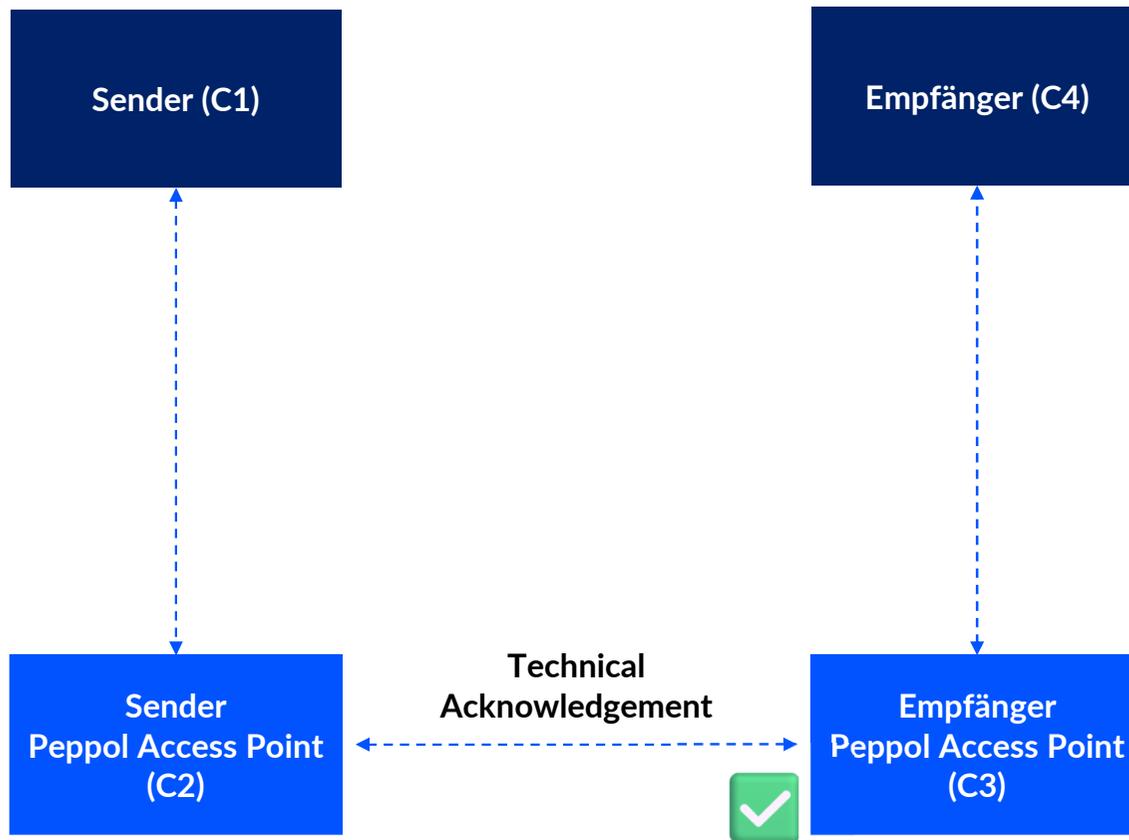
- Participant ID: 9930:de114150623
Country:  Germany (DE)
Entity Name: Fraport AG
Geographical information: 60547 Frankfurt am Main



Aufwand für das Eintragen: 10 Minuten



Bestätigungsarten bei Peppol

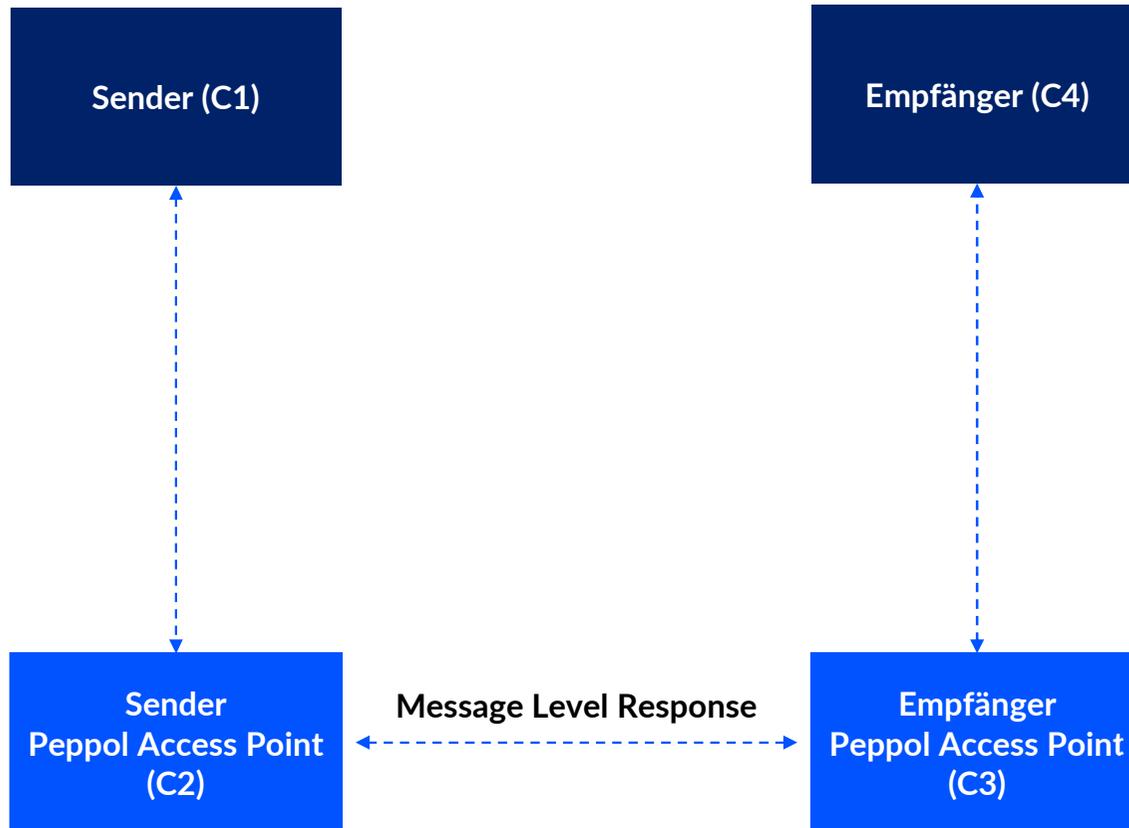


Technical Acknowledgement (Signal Message)

- Bestätigung, dass der technische Empfang am Access Point funktioniert hat
- Garantiert die Nichtabstreitbarkeit des Erhalts von C3 an C2



Bestätigungsarten bei Peppol



Message Level Response

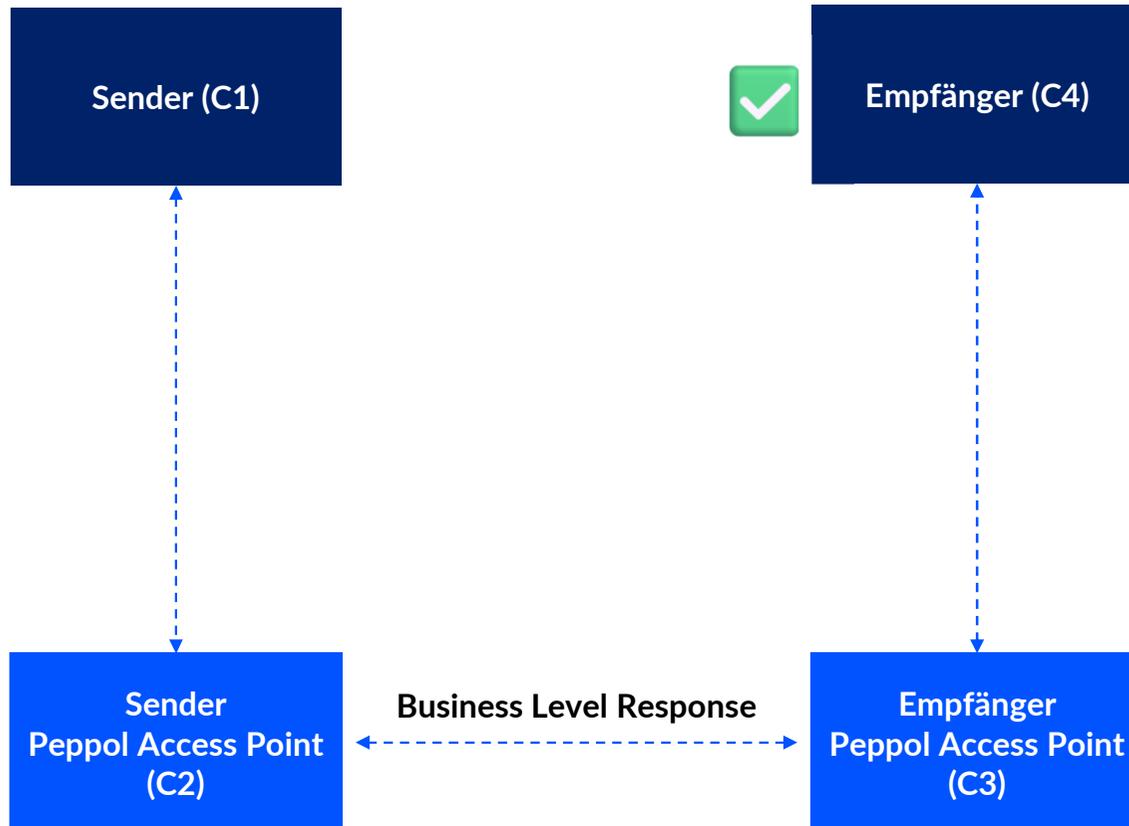
- Bestätigung, dass die Verarbeitung des gesendeten XMLs am C3 funktioniert hat
- Einer korrekten Weiterverarbeitung am C4 steht daher nichts im Wege



XML Validation passed OK



Bestätigungsarten bei Peppol



Business Level Response

- Bestätigung, dass die XML-Nachricht korrekt im Zielsystem C4 verarbeitet werden konnte
- Informiert den Sender über den Verarbeitungszustand

AB: basic message acknowledgement

AP: invoice accepted

RE: invoice rejected

IP: in progress

UQ: under query

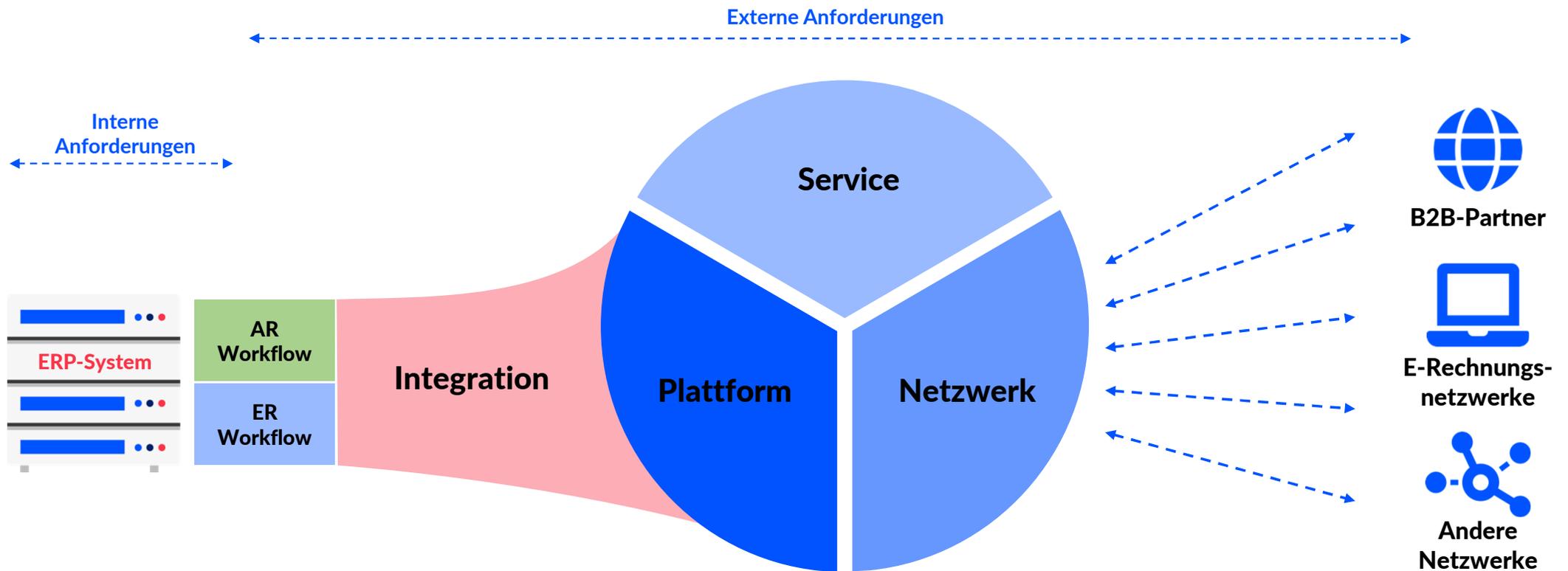
CA: conditionally accepted

PD: paid/completed

Zielarchitektur

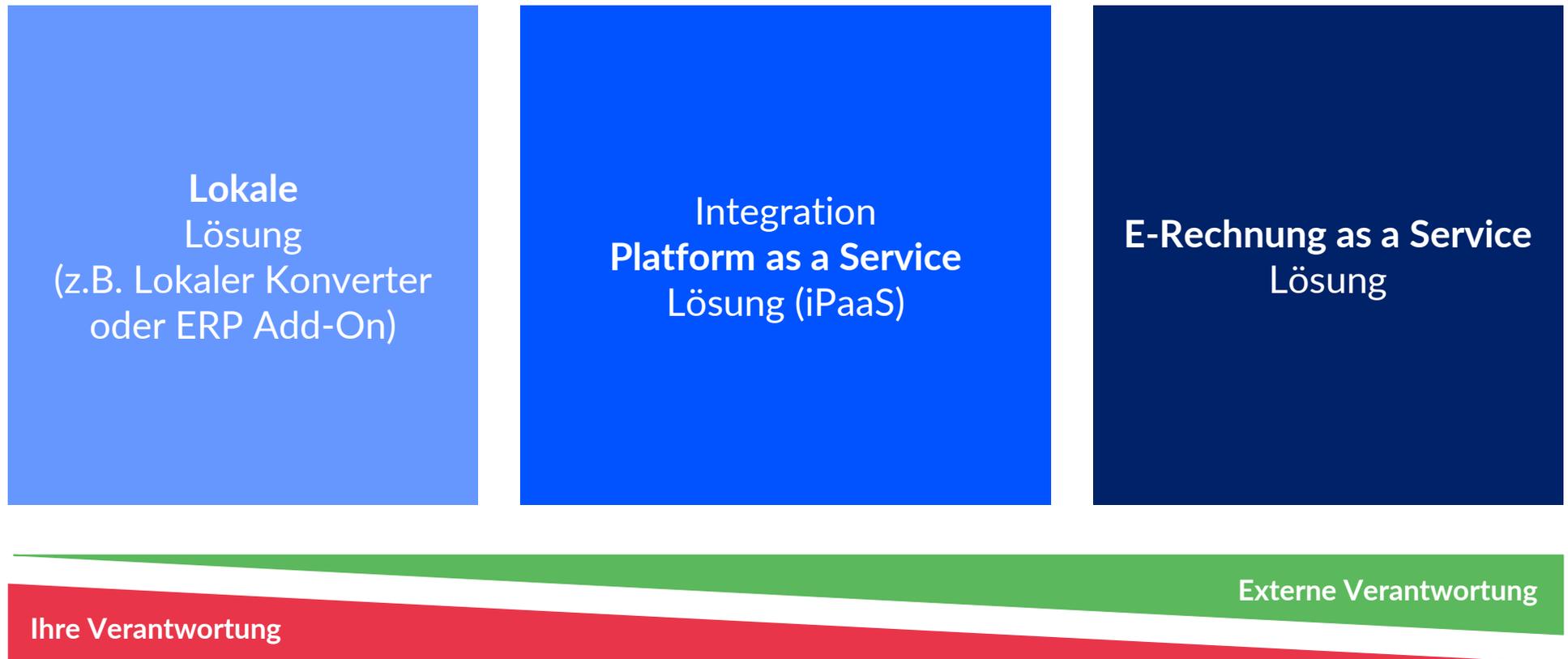


Zielarchitektur einer e-Rechnungslösung





Technische Ansätze für E-Rechnungslösungen





CONNECTIONS THAT WORK

Close to our customers



GERMANY

Leoprechtingstraße 32
81739 München
Email: edi@ecosio.com



AUSTRIA

Lange Gasse 30
1080 Wien
Email: edi@ecosio.com



UNITED KINGDOM

77 Farringdon Road
London EC1M 3JU
Email: edi@ecosio.com

