

# 中小企業共通 EDI 標準仕様書

ver.4.3\_r0\_draft\_r4\_20250901

<付属書>

構造化 CSV フォーマット仕様書

Ver.1.0\_r0

特定非営利活動法人

IT コーディネータ協会

つなぐ IT 推進協議会

共通 EDI 標準部会

#### 改定履歴

バージョン	改定	改定内容
0.0	2025/07/14	原稿作成
1.0	2025/09/01	構造化 CSV フォーマット仕様書 ver.1.0 を中小企業共通 EDI 標準 ver.4.3 の付属書として策定
1.1	2025/09/30	原稿改訂

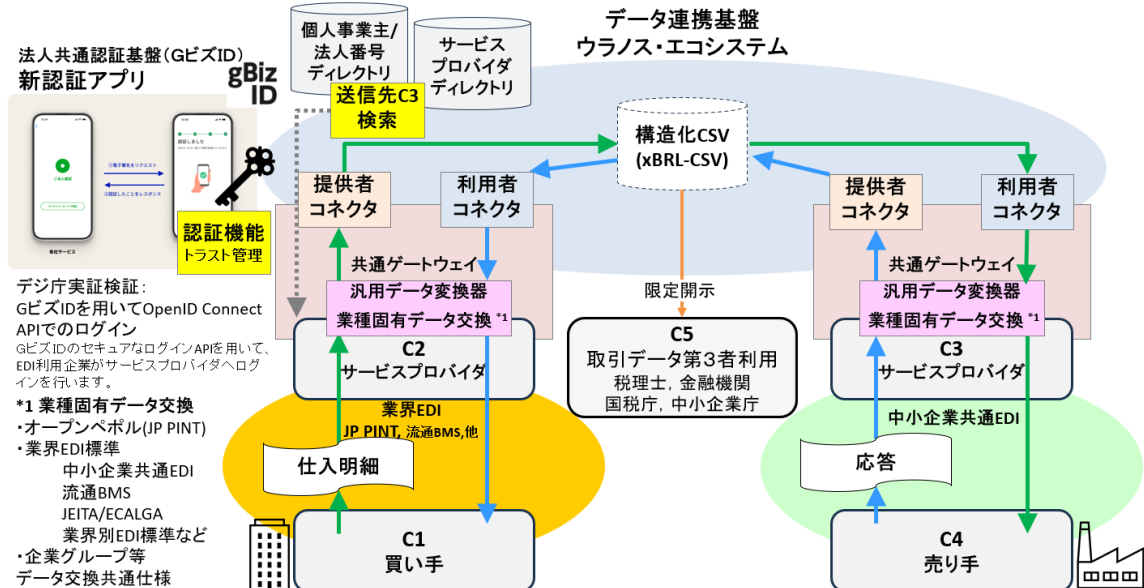
## 1. 構造化 CSV によるメッセージ定義の簡素化

電子帳簿保存法やインボイス制度の導入により、会計関連データの電子的な保存・交換の重要性が高まっています。こうした背景の中、CSV 形式のシンプルさと XML の持つ階層構造の意味付けを両立できる「構造化 CSV (xBRL-CSV)」形式が注目されています。

構造化 CSV により、ペポル JP PINT を始めとして業界 EDI や中小企業共通 EDI、電子レシート、デジタルバンキングなど様々な外部情報源から取得した会計関連データを、統一された形式で保存・活用することが可能になります。これにより、従来は複雑なデータベース管理が必要だった会計システムにおいて、システム管理の負担を軽減しながら、標準化された会計データインターフェースを実現できます。

本仕様書では、中小企業共通 EDI 標準で採用されている UN/CEFACT CCBDA 準拠の XML スキーマを基に、メッセージ定義における繰り返し構造に着目し、会計記帳に必要な情報のみを抽出して構造化 CSV を定義する方法を示します。

### データ連携基盤ウラノス・エコシステム



© 2025 SAMBUICHI PROFESSIONAL ENGINEERS OFFICE. Licensed under Creative Commons Attribution 4.0 International (CC BY 4.0).

3

### 構造化 CSV の役割

本図は、構造化 CSV が社内外のシステムをつなぐ「標準化インターフェース」として機能することを表しています。

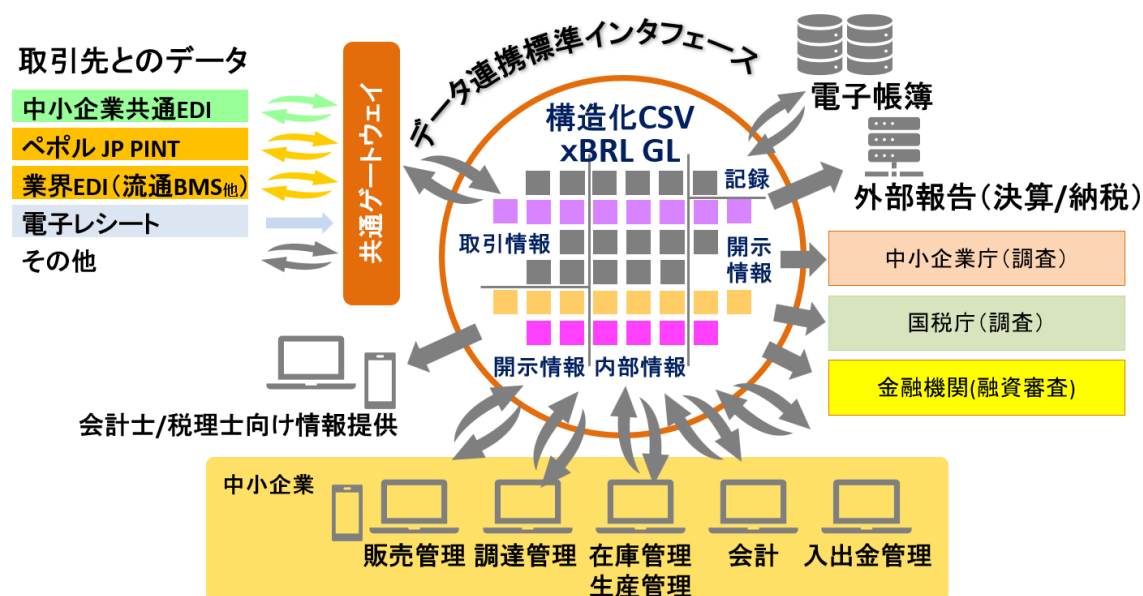
左側には、ペポル JP PINT や中小企業共通 EDI などの外部の情報源があり、これらから取得される多様な商取引データが、構造化 CSV に変換されます。

中央には、構造化 CSV が示されており、会計に必要な情報を階層的かつ統一的に保持しています。このデータは xBRL-CSV の仕組みに基づき、正確かつ再利用可能な形で表現されます。

右側には、構造化 CSV を基に各種業務システム（販売管理、調達管理、在庫管理、入出金管理）や電子帳簿、経理、外部報告（決算・納税）などに情報が提供される様子が示されています。構造化 CSV は、それらの業務間で共通言語として機能し、データの流通と連携を容易にします。

このように、構造化 CSV は、複雑な会計・業務処理の中核に位置し、異なる情報源やシステム間の整合性を保ちながら、企業の DX（デジタルトランスフォーメーション）を支える鍵となります。

### 企業間取引/会計/金融データ連携基盤ウラノス・エコシステム



© 2025 SAMBUICHI PROFESSIONAL ENGINEERS OFFICE. Licensed under Creative Commons Attribution 4.0 International (CC BY 4.0).

4

## 2. 企業間取引・決済におけるシステム連携とデータ変換基盤の構築

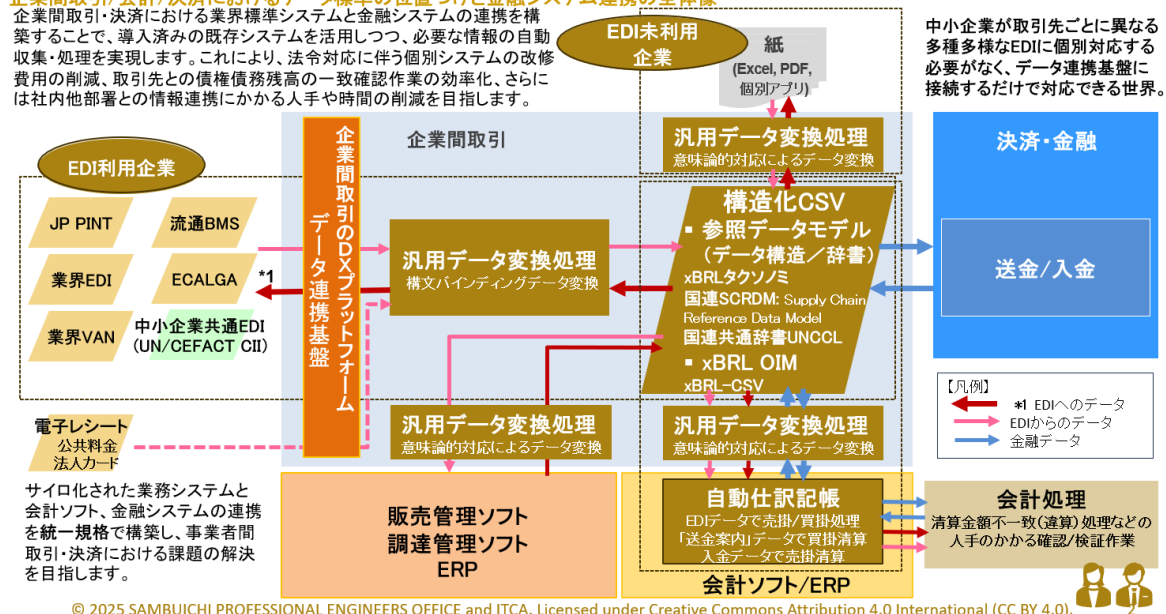
### 2.1. 概要

企業間取引や決済業務においては、EDI、電子帳簿、会計システム、金融サービスといった多様な仕組みが存在し、それぞれが独自のフォーマットや運用方法を持っています。そのため、業務の自動化や効率化を妨げる「システムの分断（サイロ化）」が大きな

課題となっています。こうした課題を解決するための共通のデータ連携基盤と汎用データ変換処理（汎用アダプター）を中核とした仕組みです。

## 企業間取引/会計/金融データ連携の全体像

### 企業間取引/会計/決済におけるデータ標準の位置づけと金融システム連携の全体像



この仕組みは、次の三層で構成されます。

### 1) 外部サービスとの接続（図左）

EDI を利用している企業は、以下のような電子商取引サービスとデータをやり取りしています：

- 中小企業共通 EDI（国際標準準拠）
- JP PINT（Peppol）
- 業界 EDI（流通 BMS、ECALGA、VAN など）
- 電子レシート、公共料金データ
- デジタルバンキング など

これらのサービスは、企業が受発注・請求・支払・入金といった業務データを電子的に交換するための基盤となります。

### 2) データ変換と連携処理（図中央）

企業間取引 DX プラットフォーム上に配置された汎用データ変換処理（汎用アダプター）は、外部・内部双方から流入するデータを以下のように統一・整形します。

- 意味論に基づくデータ変換：

異なる表記や名称（商品コード、金額の単位など）を、意味的に統一します。

- 構文／バインディング変換：

フォーマットの違い（XML, CSV, Excel 等）を吸収し、共通構造に変換します。

- 国際標準に準拠した統一構造化：

UN/CEFACT のコード体系（UNCL, UNTDID）、XBRL GL、構造化 CSV（xBRL-CSV）等に基づき、再利用可能で拡張性のある構造に整備します。

これにより、異なる EDI や紙帳票、PDF などから得られた情報も、同一の内部処理基盤上で扱えるようになります。

### 3) 社内業務・会計処理への統合（図右）

変換されたデータは、次のような企業内のシステムに自動的に連携されます：

- 販売管理、調達管理、在庫管理などの ERP

会計ソフトや仕訳記帳処理（債務計上、入出金消込、消費税対応等）

EDI を導入していない企業であっても、Excel や PDF といった紙帳票をデータ変換処理して構造化 CSV とすることで、同じインタフェースに統合され、業務・会計ソフトとの自動連携が可能となります。

## 2.2. 本仕組みによる効果と利点

- 法令対応コストの削減：

電子帳簿保存法やインボイス制度対応のための個別システム改修を最小限に抑えます。

- 債権債務の照合業務の効率化：

送金／請求データの一致確認を自動化し、人的作業の負荷を削減します。

- 社内情報の統合と再利用：

販売部門と会計部門など、異なる業務システム間の情報連携が容易になり、データの再利用や誤入力の防止に貢献します。

- 中小企業の負担軽減と DX 推進：

取引先ごとの EDI や帳票フォーマットに個別対応する必要がなくなり、一度データ連携基盤に接続すれば、あらゆるサービスとの連携が可能になります。

本図で示された構成は、「業界標準のインタフェースを介した連携」を通じて、従来のサイロ化された企業内外のデータ流通を再構成し、DX（デジタルトランスフォーメーション）を推進するための現実的なアプローチであり、中小企業にとっては、業務負担の軽減・正確な記帳・迅速な決算対応・金融機関との円滑な連携といった、実務的な導入メリットを得られる点で、非常に実効性の高い構成です。

## 3. 構造化 CSV

### 3.1. 構造化 CSV と階層型整然データ (Tidy Data)

Wickham[1]による整然データ (Tidy Data) の定義では、データは次のように整理されるべきとされています。

- **各変数は列を形成する:** データセットの各変数 (測定項目や属性) は独自の列を持ちます。
- **各観測値は行を形成する:** 各行は単一の観測値を表します。
- **それぞれの種類の観測単位が表を形成する<sup>1</sup>:** 異なる観測単位は別々のテーブルに配置されます。

構造化 CSV (xBRL-CSV[3]で定義します) は、Wickham の整然データ (Tidy Data) の概念を拡張しています。特に、階層的データの扱いにおいて次のような特徴を有しています。

- 階層的データ構造のサポート

整然データ (Tidy Data) は元々フラットなデータ構造を前提としていますが、構造化 CSV は階層的なデータ構造をサポートします。これにより、複雑な組織データや会計データのように、自然に階層化されたデータをより効率的に表現できます。

- 複数レベルの変数関連性の表現

構造化 CSV では、単一のレコード内で複数レベルの変数を関連付けることが可能です。これにより、たとえば、特定の取引に関連する会計情報や関連する事業部門の情報を、一つの統合されたビューで表現することができます。

- 拡張性と柔軟性: 構造化 CSV は、XBRL[4] タクソノミに基づいているため、既存の会計基準や報告基準に合わせて容易に拡張やカスタマイズが可能です。

---

<sup>1</sup> 例えば、仕訳帳という観測単位には、入力ごとの情報 (売掛計上、入金など)、明細行の情報 (売上や仮受消費税など)、税率別の合計といった複数の階層が含まれています。

Wickham の定義はコッド[2]の第 3 正規形の別表現であり、異なる階層を異なる観測として扱っていますが、この観測単位の解釈を拡張し、複数の階層を一つの観測対象としてまとめることで、階層的整然データ (Hierarchical Tidy Data) を定義しました。これにより、従来の関係データベースのように、文書や明細行などの階層ごとに複数の表を定義する必要がなく、一つの表で仕訳帳全体を表現できるようになります。

xBRL-CSV[3]のディメンション定義リンクベースを用いて、この階層構造を定義するのが構造化 CSV です。この仕組みにより、階層的なデータ構造を簡潔かつ効果的に表現でき、さまざまなビジネスや報告用途に対応することが可能です。

### 3.2. 仕訳情報の場合

観測単位が仕訳入力の場合、構造化 CSV は以下のような階層的なデータ表現を可能にします：

- 入力ごとの情報

各行が入力ごとの計上日付、入力日、摘要などの観測を表します。

- 明細行の情報

勘定科目ごとの金額や消費税額適用などの明細行が記録されます。

- 補助科目ごとの詳細情報

勘定科目に定義した補助科目ごとの預金口座、取引先、原価部門などの詳細情報が、同じ補助科目番号、補助科目名の欄を使って行の観測として登録されます。

- 階層的な記録

これらの階層は一つのファイルに記録でき、データの整理と分析に大きな利便性を提供します。

### 3.3. 構造化 CSV 採用の利点

- JOIN 処理の削減

一般的なリレーショナルデータベースでは、複数のテーブル間で関連するデータを結合するために JOIN 処理が必要です。しかし、構造化 CSV の採用により、関連するデータを階層的に一つのファイル内で管理できるため、このような煩雑な JOIN 処理が不要となります。

- データベースの維持管理が不要

リレーショナルデータベースの維持管理は、データの整合性、セキュリティ、バックアップなど多くの作業が伴います。しかし、構造化 CSV を用いることで、これらの複雑なデータベース管理作業が不要となり、管理の負担が大幅に軽減されます。

- プログラム処理の単純化と高速化

構造化 CSV を用いることで、データの処理が単純化され、高速化が可能になります。データが一つのファイル内で完結しているため、データの読み込みや書き込みが効率的に行われ、全体のデータ処理の速度が向上します。

- まとめ

構造化 CSV の採用は、データ処理の効率化に大きく寄与しています。これにより、XML や JSON のような階層構造を持つデータを一つのファイルで表現でき、煩雑な



JOIN 処理が不要になります。これらの拡張により、構造化 CSV は Wickham の Tidy Data の基本的な原則を維持しつつ、より複雑で階層的なデータ構造を扱う現代のビジネスや会計のニーズに対応しています。これにより、データの分析、解釈、報告が容易になり、組織の意思決定プロセスを支援します。また、リレーショナルデータベースの管理負担の軽減、データ処理の単純化と高速化が実現され、企業のデータ管理と分析がより効率的かつ迅速に行えるようになります。

## 4. 中小企業共通 EDI 標準構造化 CSV フォーマットについて

### 4.1. 概要

中小企業共通 EDI 標準のメッセージ定義は、UN/CEFACT の CCBDA ルールに従って作成された XML スキーマを基礎としている。構造化 CSV では、この XML スキーマ定義における complexType 定義のうち、maxOccurs="unbounded"とされている要素のみを新たな階層レベルとして扱い、それ以外は同一レベルまたは非表示（対象外）とすることで、論理階層構造を維持しつつ、CSV 形式に落とし込む手法である。

### 4.2. 対象外とする要素

以下の要素は構造化 CSV 定義から除外する：

maxOccurs="1" の complexType 要素（階層を深くしない）

制御情報やヘッダー項目など、メッセージ交換の管理に関わる項目

会計記録に直接関係しない補足的な情報（例：送信方法 など）

### 4.3. インボイス文書の例

XML スキーマ：

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://example.org/invoice"
elementFormDefault="qualified"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:inv="http://example.org/invoice">
<!-- ルート要素 Invoice -->
<element name="Invoice">
  <complexType>
    <sequence>
```

```

<!-- Header (maxOccurs=1) -->
<element name="Header">
  <complexType>
    <sequence>
      <element name="InvoiceNumber" type="string"/>
      <element name="InvoiceDate" type="date"/>
      <element name="Seller">
        <complexType>
          <sequence>
            <element name="Name" type="string"/>
            <element name="TaxID" type="string"/>
          </sequence>
        </complexType>
      </element>
      <element name="Buyer">
        <complexType>
          <sequence>
            <element name="Name" type="string"/>
          </sequence>
        </complexType>
      </element>
    </sequence>
  </complexType>
</element>
<!-- LineItem (maxOccurs="unbounded") -->
<element name="LineItem" maxOccurs="unbounded">
  <complexType>
    <sequence>
      <element name="LineNumber" type="string"/>
      <element name="Item" type="string"/>
      <element name="Amount" type="decimal"/>
    </sequence>
  </complexType>
</element>
</sequence>

```

```
</complexType>
</element>
</schema>
```

注：XML スキーマ定義では、minOccurs および maxOccurs の規定値は"1"です。

XML 文書：

```
<Invoice>
  <Header> <!--maxOccurs="1"：階層を深くしない -->
    <InvoiceNumber>INV001</InvoiceNumber>
    <invoiceDate>2025-05-17</invoiceDate>
    <Seller>
      <Name>■■商事</Name>
      <TaxID>T012345678901</TaxID>
    </Seller>
    <Buyer>
      <Name>株式会社〇〇</Name>
    </Buyer>
  </Header>
  <LineItem> <!--maxOccurs="unbounded"：階層レベルを深くする -->
    <LineNumber>INV001-1</LineNumber>
    <ItemName>商品 A</ItemName>
    <Amount>1000</Amount>
  </LineItem>
  <LineItem>
    <LineNumber>INV001-2</LineNumber>
    <ItemName>商品 B</ItemName>
    <Amount>2000</Amount>
  </LineItem>
</Invoice>
```

## 5. 対応定義表の作成手順

### 5.1. XML スキーマの解析

上記例のスキーマ定義からスキーマ定義表を作成します。

表 1－XPath 定義表

No.	Level	Name	Type/Datatype	maxOccurs	XPath
1	0	Invoice	complexType		/Invoice
2	1	>Header	complexType		/Invoice/Header
3	2	>>InvoiceNumber	String		/Invoice/Header/InvoiceNumber
4	2	>>InvoiceDate	date		/Invoice/Header/InvoiceDate
5	2	>>Seller	complexType		/Invoice/Header/Seller
6	3	>>>Name	string		/Invoice/Header/Seller/Name
7	3	>>>TaxID	string		/Invoice/Header/Seller/TaxID
8	2	>>Buyer	complexType		/Invoice/Header/Buyer
9	3	>>>Name	string		/Invoice/Header/Buyer/Name
10	2	>>LineItem	complexType	unbounded	/Invoice/LineItem
11	3	>>>LineNumber	string		/Invoice/LineItem/LineNumber
12	3	>>>ItemName	string		/Invoice/LineItem/ItemName
13	3	>>>Amount	decimal		/Invoice/LineItem/Amount

complexType 構造を解析し、maxOccurs 属性値を確認するとともにその要素の文書中の位置を指定する XPath を確認します。

## 5.2. 構造化 CSV の階層定義ルール

- スキーマ中の maxOccurs="unbounded" をもつ complexType で定義された element に注目して階層化する。complexType で定義された element は sequence に含まれる他の udt や qdt の element と同じ階層として定義する。complexType の下位の element は、階層を+1する。
- complexType が maxOccurs="1" の complexType は対象外とする。その下位の element は、階層を深くすることなく同一階層として登録する。

### 2.2 対応定義表の作成規則

対象 XML 要素を一覧化し、CSV カラムとの対応を定義する。

対象外要素は除外する。

XPath やレベル番号、論理モデルの項目名も定義に含める。

構造化 CSV のカラム名は、論理モデルの項目名から別途定める命名規則によって生成する。

表 2－構造化 CSV 対応定義表

階層：dInvoice → dInvoiceLine

InvoiceLine を繰返しとして CSV の複数行に展開

請求番号や取引先などのヘッダー項目はインボイス文書の階層に XML スキーマの定義する階層に関係なく統合表示する。

dInvoice,dInvoiceLine,invoiceNumber,invoiceIssueDate,SellerName,SellerTaxIdentifier,BuyerName,InvoiceLineID,InvoiceLineNetAmount,ItemName
1,,INV001,2025-06-01,株式会社〇〇,■■■商事,T012345678901,,
1,1,,,,,INV001-1,1000,商品 A
1,2,,,,,INV001-2,2000,商品 B

他形式との比較

項目	構造化 CSV	XML	JSON
階層表現	○（レベルで管理）	○（レベルで管理）	○（レベルで管理）
可読性	◎（表形式）	△	△
機械処理性	◎	◎	◎
ファイルサイズ	小	大	中
標準規格との対応	○xBRL-CSV 準拠	◎UN/CEFACT 準拠	○JSON Schema 定義

## 6. ツールによる CSV 自動生成手順

- 構造化 CSV 対応定義表を取得
- アプリのデータモデルを論理階層モデルに変換
- Python スクリプトを用いて、論理階層モデルを構造化 CSV に変換
- Python スクリプトを用いて、構造化 CSV を論理階層モデルに変換
- 論理階層モデルをアプリのデータモデルに変換

上記のデータ変換ステップ c)、d) は、オープンソースのプログラムで処理が提供されます。この変換処理を参考に、それぞれのアプリのデータ構造に応じて、ステップ a)、e) の処理を XPath を参照する変換処理に変更すれば、将来的なアプリの更新に伴う物理ファイルフォーマットの変更や新たな EDI サービスに対応するときにも、構造化 CSV 対応定義表の差し替えのみで対応可能となり、プログラムの修正は不要になります。

## 7. 参考文献

- [1] Tidy Data, Journal of Statistical Software, August 2014, Volume 59, Issue 10, Headley Wickham  
<https://www.jstatsoft.org/article/view/v059i10>
- [2] E. F. Codd, Further Normalization of the Data Base Relational Model, IBM Research

- Report, San Jose, California RJ909 , August 31, 1971 [viewed 2022-09-29]. Available at <https://forum.thethirdmanifesto.com/wp-content/uploads/asgarosforum/987737/00-efc-further-normalization.pdf>
- [3] xBRL-CSV: CSV representation of XBRL data 1.0, Recommendation 13 October 2021, XBRL International Inc.  
<https://www.xbrl.org/Specification/xbrl-csv/REC-2021-10-13/xbrl-csv-REC-2021-10-13.html>
- [4] Extensible Business Reporting Language (XBRL) 2.1, Recommendation 31 December 2003 with errata corrections to 20 February 2013, XBRL International Inc.  
<https://www.xbrl.org/Specification/XBRL-2.1/REC-2003-12-31/XBRL-2.1-REC-2003-12-31+corrected-errata-2013-02-20.html>
- [5] AICPA, Audit Data Standards General Ledger Standard, [viewed 2023-03-29]. Available at  
<https://us.aicpa.org/content/dam/aicpa/interestareas/frc/assuranceadvisoryservices/downloadabledocuments/auditdatastandards/auditdatastandards.gl.july2015.pdf>
- [6] XBRL GL Taxonomy Framework Technical Architecture 2015 Recommendation 25 March 2015, [viewed 2023-08-26]. Available at <https://www.xbrl.org/int/gl/2015-03-25/GLTFTA-REC-2015-03-25.html#section-control-over-the-modification-of-content-models>
- [7] XML Schema Part 2: Datatypes Second Edition  
<https://www.w3.org/TR/xmlschema-2/>