PDF/A

Inline XBRL
and
Report Package

Herm Fischer
Exbee Ltd

DIGITAL REPORTING IN EUROPE
3-5 JUNE 2025, FRANKFURT
Hosted by the European Central Bank

eurofiling

xBRL | EUROPE

PDF/A is called for

- PDF/A is an ISO standard (since 2017)
    - Managed by pdfa.org as an open public standard
    - Includes structural tagging
- FDTA identifies PDF/A as an XBRL alternative
- PDF/A can be both:
    1. An HTML rendering alternative for inline XBRL
    2. A Report Package alternative for zip files

History of PDF for XBRL

- Early 2000's – Proprietary and licensing, model difficulties
- 2017 – became an ISO standard with a structural model
- 2024
  - Sept XTAC, XSB interest in PDF for inline
  - Nov Madrid meeting proof of concept demos
- 2025
  - PDF week in Boston – evolve approach with community
  - Summer – evolve proof of concept to community approach
  - PDF week in Berlin…

Use Case Problem Statements

1. Inline XBRL Reports (top priority)
    - Preparation in PDF and conversion to HTML unsatisfactory
        - Absolute positioning vs reflow technologies unreconcilable
        - Unsatisfactory large-filing performance
    - Report packages need a tool (or website) to be viewed
2. Static forms (a costs issue)
    - SEC website GUI to prepare, style sheets viewing, etc
    - EU ESG filings (static form vs custom inline XBRL)
3. Large filings of Inline XBRL face statements + many tables
    - Filer costs extreme to tag tables in inline XBRL
    - Viewability issues (specialized HTML required for tables)
    - Doubtful usefulness of inconsistently tagged tables

PDF/A as HTML rendering alternative

- Modern PDF tools add accessibility structure
  - Visually (such as with Acrobat)
  - Online (such as Adobe, PDFIX)
  - Programmatically (such as Python pdfix autotag of tables)
- PDF structure maps to facts with inline features (tranforms, scale)
- PDF architecture supports
  - Pixel perfect rendering
  - Huge documents (page-at-a-time)

Proof of concept demo

- Draft POC Arelle plugin
  - PDF/A with structure (added by Acrobat)
  - Instance template (e.g. OIM-JSON with pdfIdRefs to structure)
  - Outputs
    - OIM-JSON with values from PDF text and fields
    - PDF/A ixviewer metadata for Arelle inline viewer
- Draft POC Arelle inline viewer
  - Use pdf.js to view/markup PDF in browser
  - (in progress)
- Work in progress: POC XBRL plugins to PDF tools

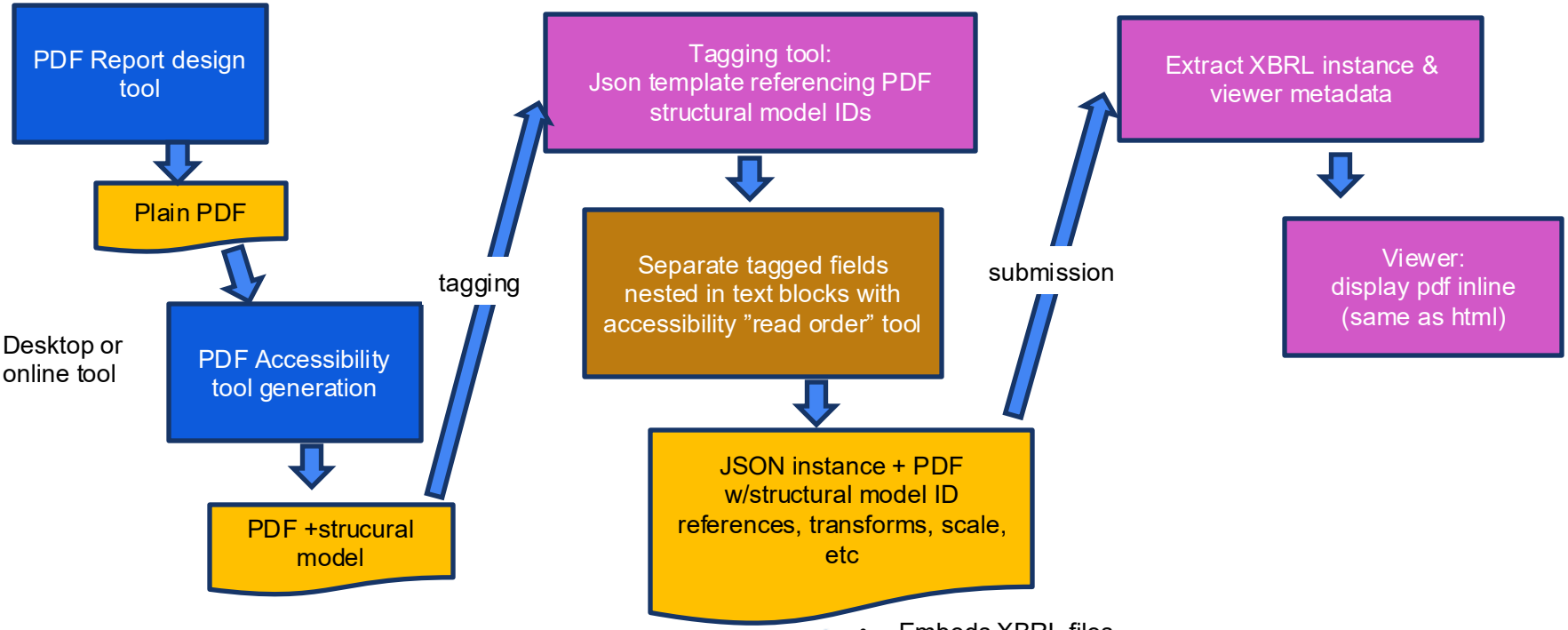
Tagging tools

- Manual tagging
  - POC requires structure model and used ID assignment/editing
    - Transitioning to iXBRL attributes on structural model elements
  - PDF editing tools do manual association of XBRL attributes to facts
    - Transitioning to POC Plugins for Acrobat and Foxit
- Autotagging
  - Python pdfix-autotag is a learning model-based table autotagger

Use Case 1 – Inline XBRL Reports

- Financial statements
- Authority documents
- ESG statements

Initial investigation of post-production tagging

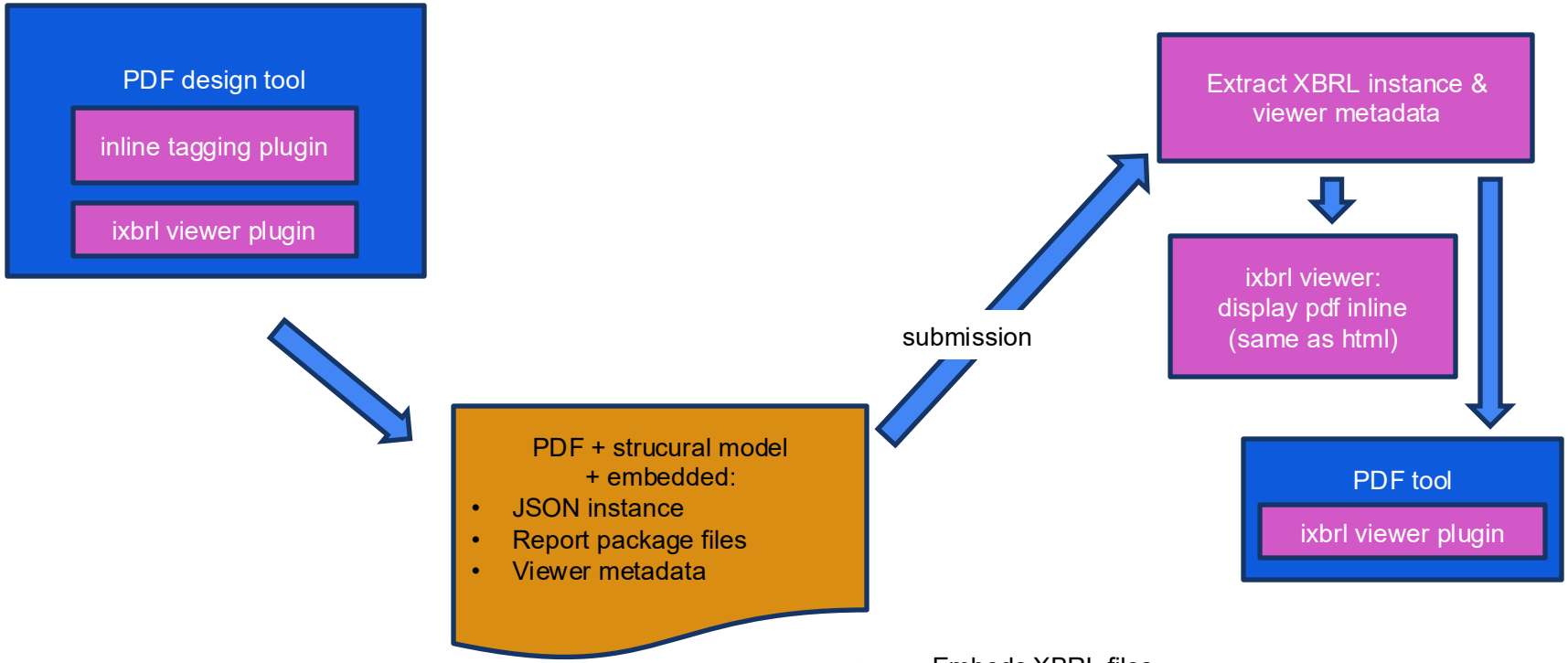


- Desktop or online tool
- Embeds XBRL files

eurofiling

XBRL EUROPE

Current direction: tagging at authoring stages



- Embeds XBRL files

eurofiling xBRL EUROPE

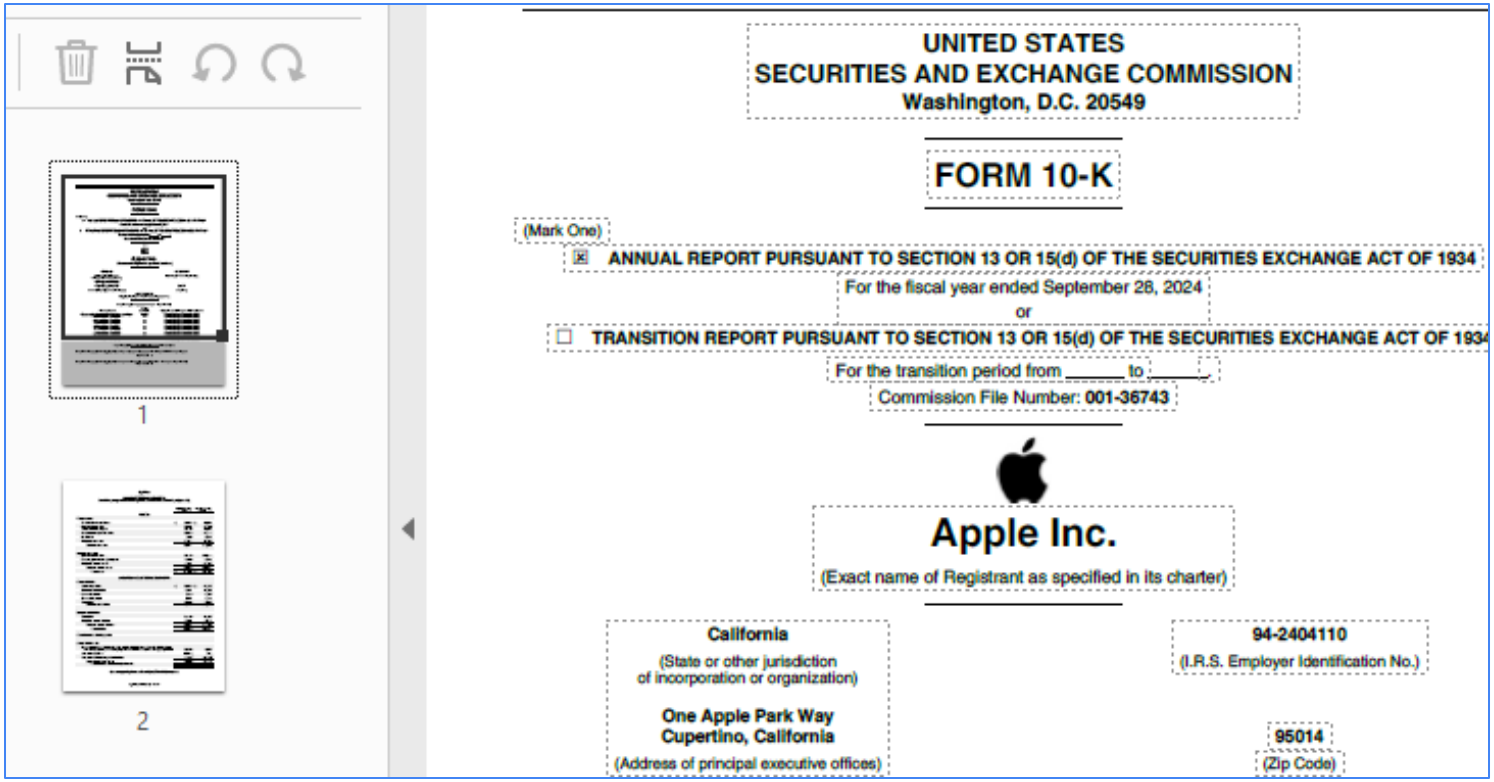
Apple 10-K example experience

- Cover page and balance sheet (other pages deleted)
- Tried multiple editors on pdf and multiple tagging tools
- Editors (to cut down to cover and balance sheet)
  - Mac editor: glyphs displayed, totally messed up text fields
  - Adobe DC: preserved text fields as expected
- Taggers
  - Adobe DC: lost glyph viewability on page 2 tables, good structure
  - Adobe Online: preserved viewability but table cells merged vertically
  - PDFIX Online: useful table cells and viewability but no element IDs

Apple 10-K

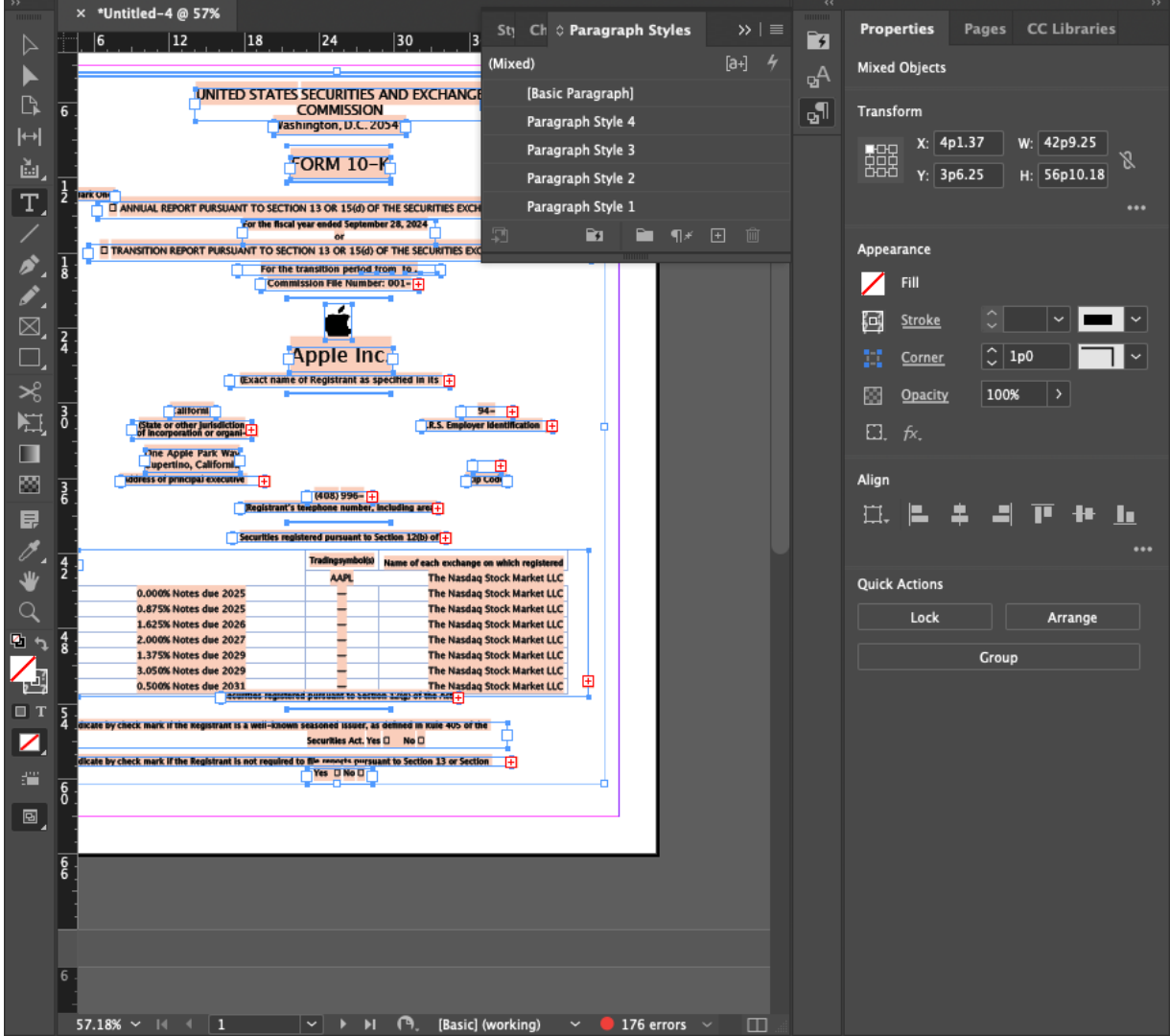
- Front page
- Balance sheet

(rest omitted)



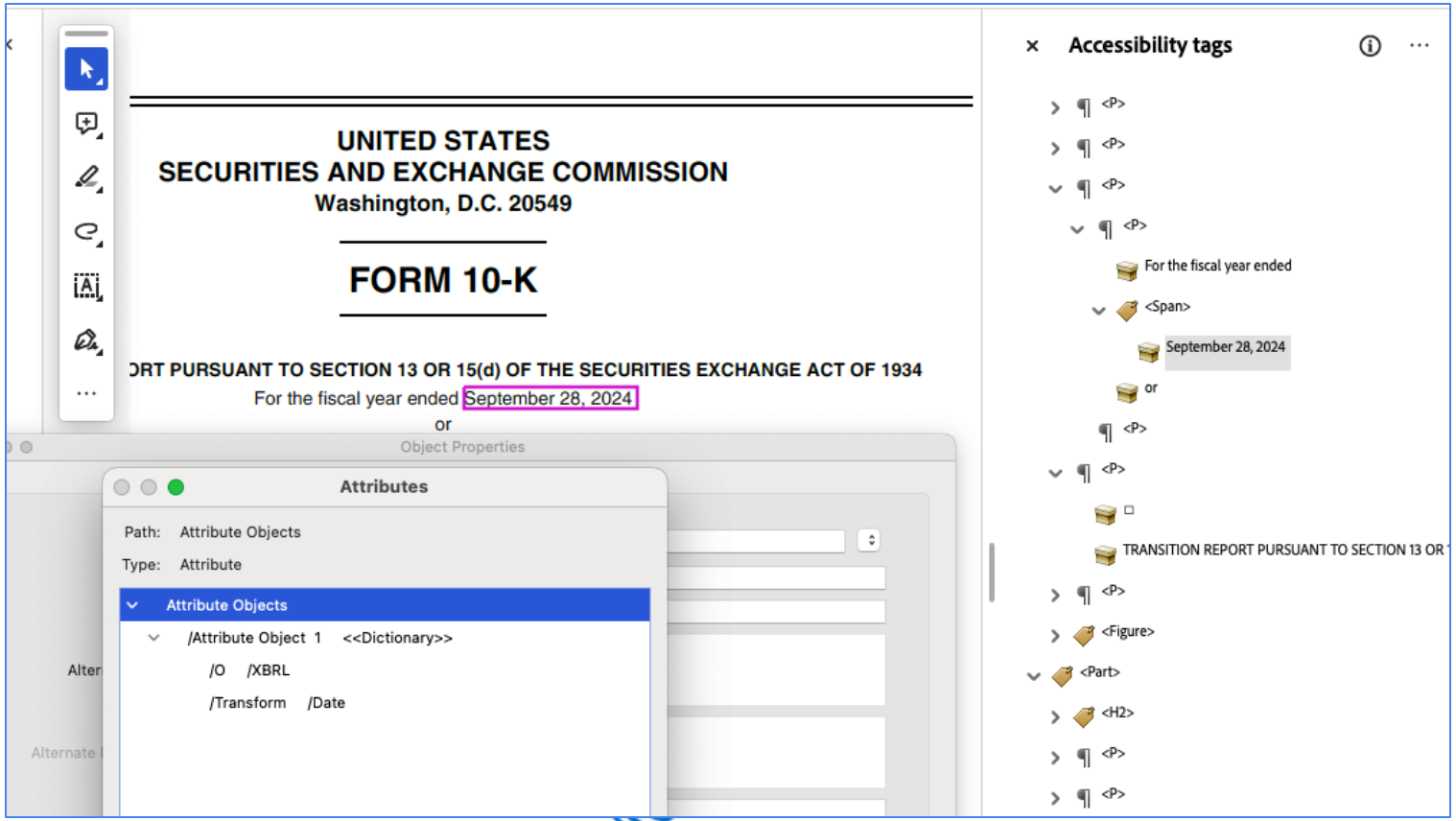
InDesign

- Photoshop-like UI
- Fixed styles on
  - Paragraph
  - Character group
- Exports to Acrobat for tagging
- No facility to capture XBRL tagging?



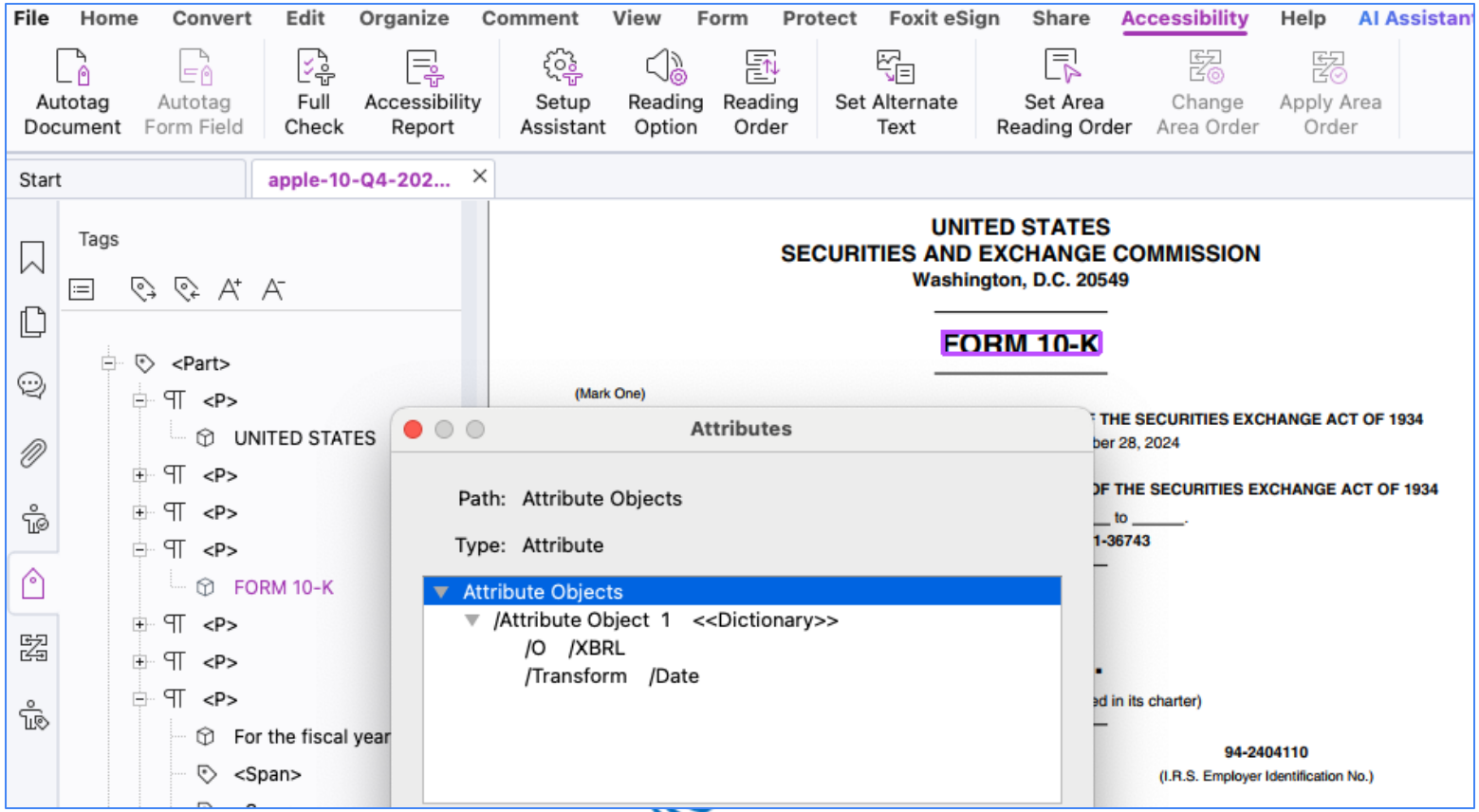
Acrobat

- Tag and tree editing
- Attributes editing
- Requires plugin for reasonable
  - Entry
  - Viewing

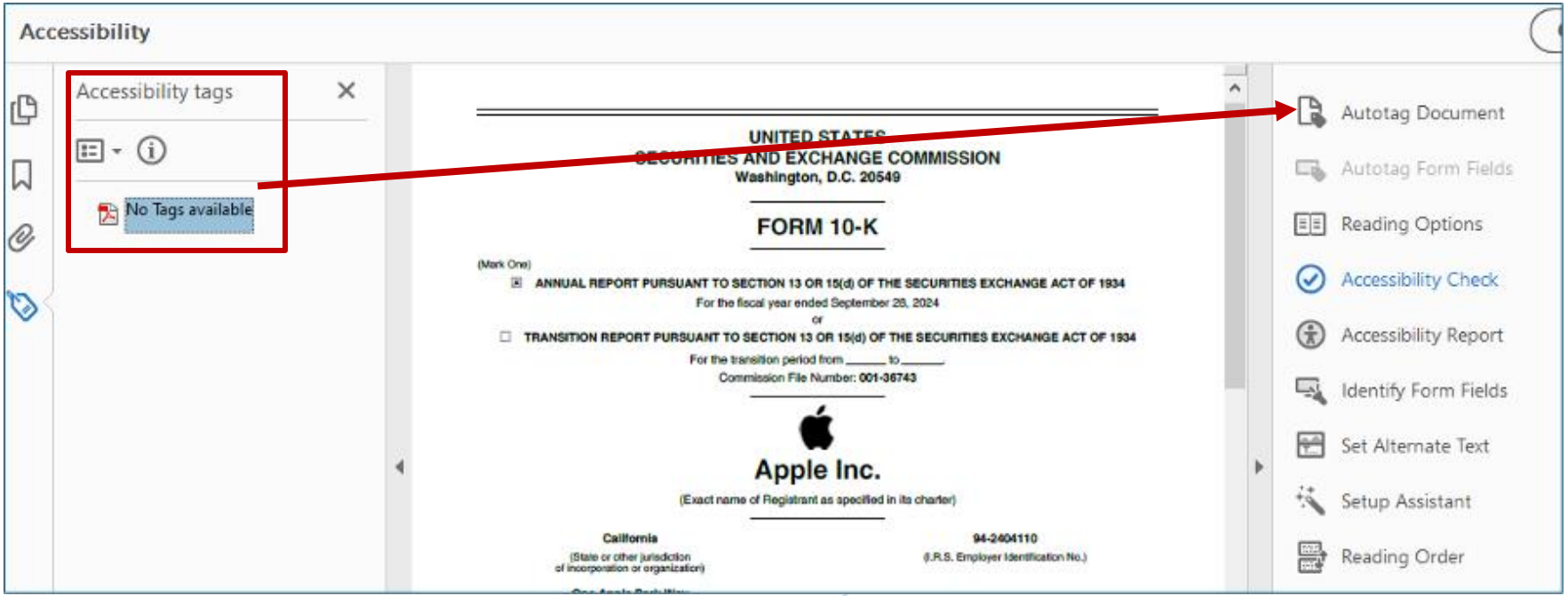


Foxit

- Tag and tree viewing only
- Attributes editing
- Requires plugin for reasonable
  - Entry
  - Viewing



Acrobat:
Show/Hide -> navig panes -> accessbility
View -> tools -> accessibility



XBRL and metadata is embedded in PDF/A

- xBRL-JSON facts reference PDF structural elements with inline-XBRL markup features (transformation, scaling, sign)
- Values of facts are obtained from PDF/A displayed text (or form field entries)
- Viewer metadata will also be embedded

PDF viewer ⇒ struct element ⇒ XBRL template

CONSOLIDATED BALANCE SHEETS

(In millions, except number of shares, which are reflected in thousands, and par value)

	September 28, 2024	September 30, 2023
ASSETS:		
Current assets:		
Cash and cash equivalents	$ 29,943	$ 29,965
Marketable securities	35,228	31,590
Accounts receivable, net	33,410	29,508
Vendor non-trade receivables	32,833	31,477
Inventories	7,286	6,331
Other current assets	14,287	14,695
Total current assets	152,987	143,566

pdf_43: 29,943
pdf_44: $
pdf_45: 29,965
pdf_46: Marketable se
pdf_47: 35,228
pdf_48: 31,590
pdf_49: Accounts rece
pdf_50: 33,410
pdf_51: 29,508
pdf_52: Vendor non-tra
pdf_53: 32,833
pdf_54: 31,477
pdf_55: Inventories
pdf_56: 7,286
pdf_57: 6,331


```
"f-147": {
  "pdfIdRefs": "pdf_43",
  "pdfFormat": "ixt:num-dot-decimal",
  "pdfScale": 6,
  "decimals": -6,
  "dimensions": {
   "concept": "us-gaap:CashAndCashEquiv…",
   "entity": "cik:0000320193",
   "period": "2024-09-29T00:00:00",
   "unit": "iso4217:USD"
  }
 },
 "f-148": {
  "pdfIdRefs": "pdf_45",
  "pdfFormat": "ixt:num-dot-decimal",
  "pdfScale": 6,
  "decimals": -6,
  "dimensions": {
   "concept": "us-gaap:CashAndCashEquiv…",
   "entity": "cik:0000320193",
   "period": "2023-10-01T00:00:00",
   "unit": "iso4217:USD"
  }
 },
```


eurofiling

Proof–of-concept implementation

- Arelle plugin
  - Integrated operation: load from PDF with JSON template
    - Template and xsd's may be embedded or separate
    - Produces instance and viewer metadata
  - Stand alone operation:
    - Trace document info, marked text and structural tagged IDs
    - Embedded or separate template and xsd's
    - Produces output or embedded JSON instance (populated from text and form fields)
- Proof of concept viewer
  - Load PDF and JSON output w/viewer metadata
  - Highlights text and table cells mapped to facts
  - Click mapped text and table cells pops up corresponding JSON instance syntax

Inline Viewer accommodation of PDF

- UX unchanged
  - PDF display overlay canvas for tagged fact outlines and mouse actions
  - Keep current Viewer UX, treat facts as tagged PDF text and form entries
  - Replace HTML DOM with pdf.js
    - Mozilla foundation
    - Apache 2.0 license
    - Supports PDF object model access needed for structural model

PDF tools accommodation of PDF

- UX unchanged
  - Add plugin with tag-entry and XBRL-properties panes
    - (Adopt from iXBRL Viewer UI?)
  - PDF-editor display features
    - tagged fact outlines
    - Structural model XBRL attributes (fact ID, transform, scaling, sign, ftnote)
    - XBRL fact. properties pane

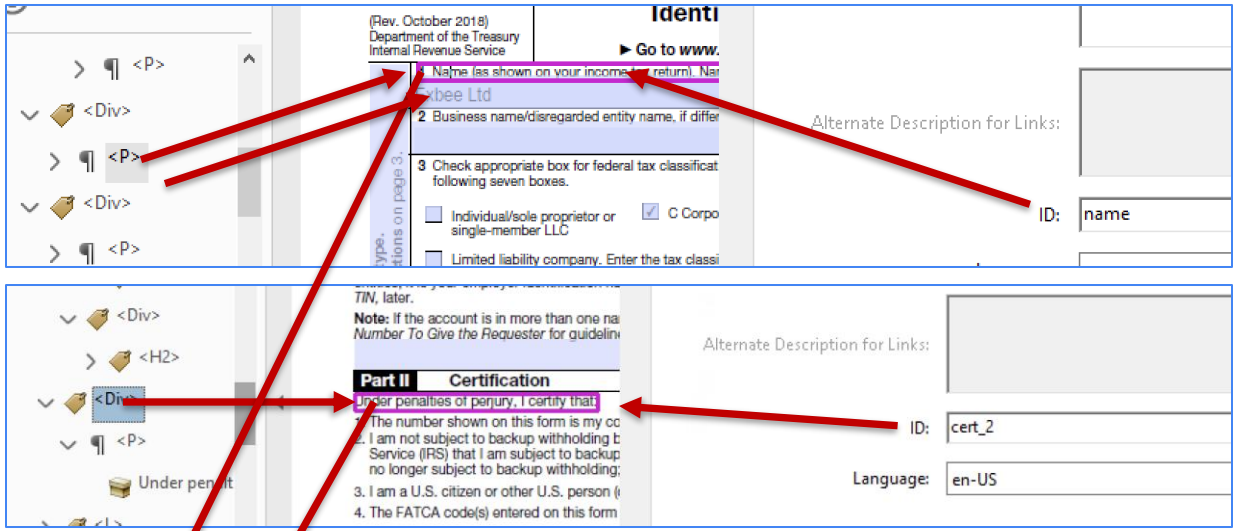
PDF/A as a Report Package container

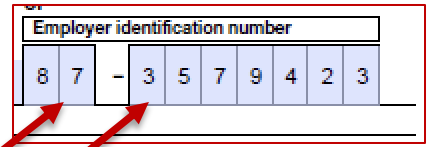
- PDF Portfolios (Adobe) aka Portable Collections (ISO)
  - Acts like a zip file with pdf suffix and view functions
  - Embed Report Package in Portfolio directory structure
  - Open in viewing tools as first report
  - Portfolios may be a seldomly-used feature (investigate?)
- Plain non-Portfolio PDF report packages
  - Flat file structure - emulate directory structure in file names (like zip)
  - Avoid concern of seldomly-used PDF feature?

Use Case 2 – Form templates for end users

- Static forms can be PDF/A with XBRL structural tagging
  - Users fill in form which is then a viable inline-XBRL report
  - Replaces need for users to have access to inline XBRL preparation tools
- Authority static forms
  - Currently XML + style sheet + custom GUI programming
  - Replace with simple PDF/A form with inline tagging
- ESG for small company reporting (EU, Calif)
  - Simple PDF/A form with static tagging

Structure model



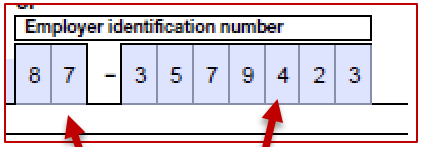



```
text blocks:
name: 1  Name (as shown on your income tax retu
business: 2  Business name/disregarded entity na
address: 5  Address (number, street, and apt. o
cityStZip: 6  City, state, and ZIP code
ssn: Social security number
cert_1: Part II
Certification
cert_2: Under penalties of perjury, I certify th
```

form fields:
f1_1[0]: Exbee Ltd
f1_7[0]: 16400 Ventura Blvd Ste 326
f1_8[0]: Encino, CA 91436
f1_9[0]:
f1_14[0]: 87
f1_15[0]: 3579423

eurofiling
xBRL EUROPE

Template produces facts from PDF/A



form fields:
f1_1[0]: Exbee Ltd
f1_7[0]: 16400 Ventura Blvd Ste 326
f1_8[0]: Encino, CA 91436
f1_9[0]:
f1_14[0]: 87
f1_15[0]: 3579423

OIM-JSON template

```
"f8": {
    "pdfldRefs": "f1_14[0] f1_15[0]",
    "dimensions": {
      "concept": "concept:ssn",
      "entity": "scheme:01",
      "period": "2024-01-
01T00:00:00"
    }
  },
```

OIM-JSON instance

```
"f8": {
    "dimensions": {
      "concept": "concept:ssn",
      "entity": "scheme:01",
      "period": "2024-01-
01T00:00:00"
    },
    "value": "87 3579423"
  },
```

eurofiling

xBRL EUROPE

Template files

- Syntax alternatives
  - Legacy inline 1.1 xhtml template instead of JSON template?
  - Cube-based facts model in JSON or CSV?
- Proof-of-concept features
  - Get attributes from structural text fields and form fields
    - Transform, scale and sign features for numeric facts
  - Use value in template when structural text fields not suitable
    - E.G. not carved up small enough, images, etc
  - Generated references for structural text when absent from tagging tool

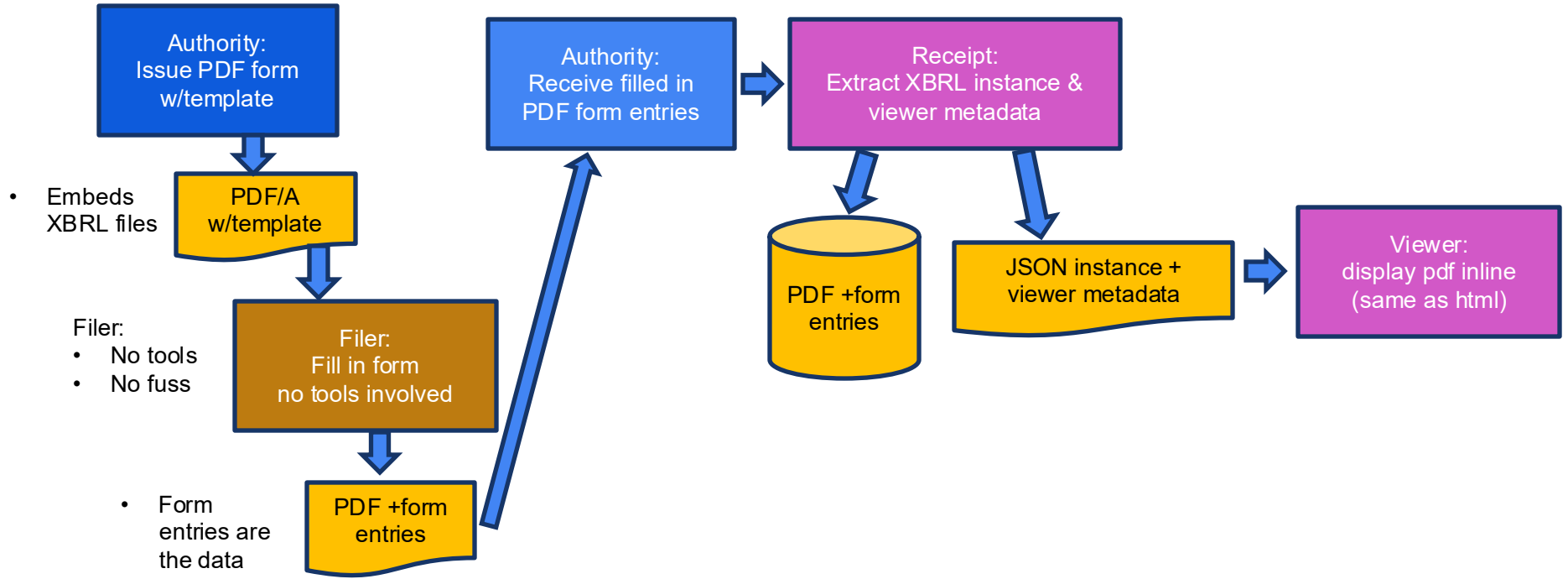
Is a separated template ‘file’ a good idea?

- Inline (xhtml)
  - 1.1 locates iXBRL features within each rendered element
  - 2.0 plans to embed iXBRL in a JSON object
- PDF/A
  - POC locates iXBRL features in a JSON payload
  - PDF/A would allow embedding xbrl features in each structural element
    - Tooling more involved when dispersed to structural elements
    - Structural edits (such as to regroup text features in accessibility editing) would require special tooling to move embedded xbrl features to regrouped structural nodes whereas separate payload only cares about attributes on structural element or its ancestor structural element

Simple static forms proposal

- Use cases:
  - SEC: forms with fixed fields and no extension taxonomy (SDR application)
  - EU ESG: small company disclosures with static ~50 elements report
- No tools or software required:
  - Authority distributes pre-tagged PDF/A form including XBRL template
    - No web form GUIs, no style sheets, no expense
  - Filer fills in and submits form
    - No filer inline XBRL tool, no tagging, just fill PDF form in and submit
  - EDGAR receives and sends PDF thru Arelle as an inline instance
    - Display w/open-source viewer enhanced for PDF in addition to HTML reports

Static forms proposal



- Embeds XBRL files

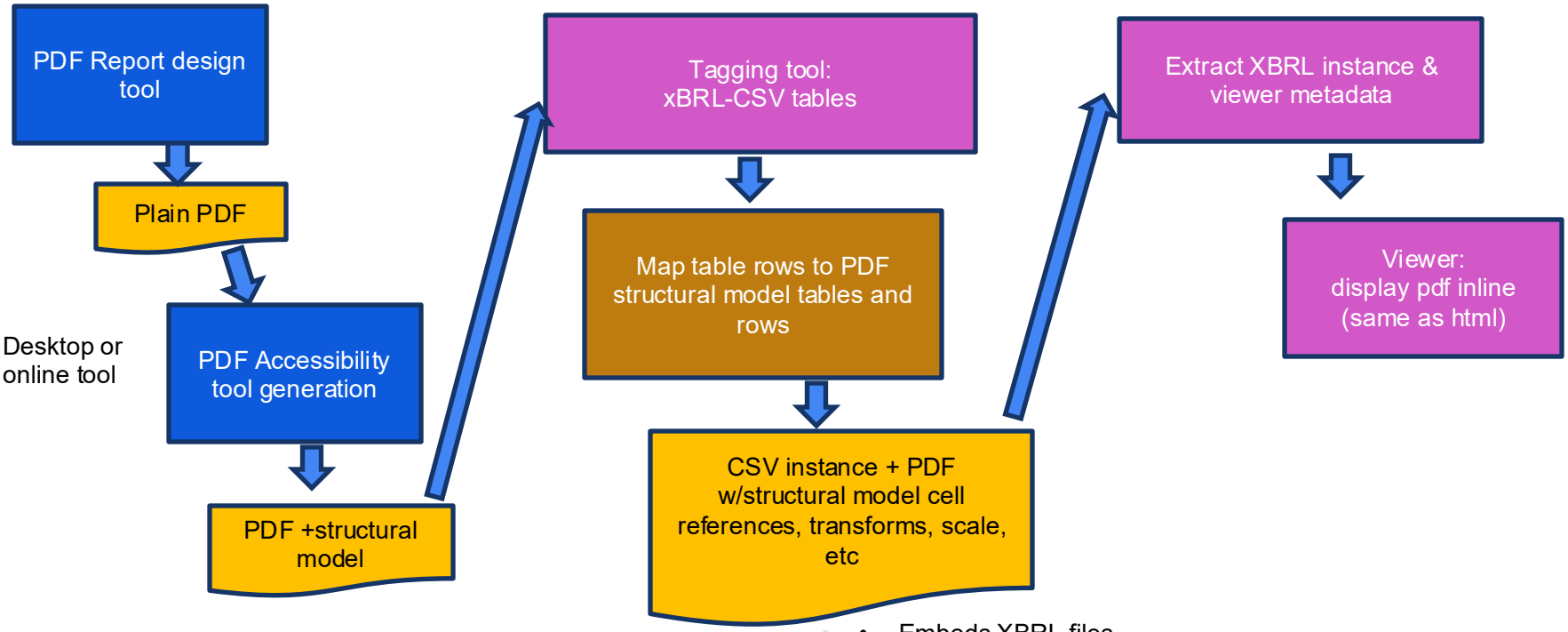
Filer:
- No tools
- No fuss

- Form entries are the data
- XBRL files are embedded

Use Case 3 – tagged tabular data

- Large filings contain face statements + huge tabular tables
  - Filer burden to tag tables with inline XBRL
  - Tables are not consumable as presently tagged (N-CSR)
- PDF/A can combine
  - Face statements (tagged the usual way)
  - Tabular data (xBRL-CSV from Excel source) row tagged
    - Serious savings to filers
    - Consummable XBRL data

Tabular submission proposal



- Desktop or online tool

- Embeds XBRL files

eurofiling

xBRL EUROPE

Thanks for your attention

DIGITAL REPORTING IN EUROPE

3-5 JUNE 2025, FRANKFURT

Hosted by the European Central Bank

eurofiling

xBRL EUROPE