29th Eurofiling Conference

5th and 6th June 2024 | Central Bank of Ireland | Dublin, Ireland

Using Arelle for ESRS

Tom Wacha, Arelle
Herm Fisher, Arelle

Arelle Overview

- Arelle is a free and open source XBRL platform that was created in 2010 to improve the accessibility and usability of XBRL in order to increase worldwide adoption of the standard.

- Our vision is to be the free, open source platform that enables regulators, auditors and filers to build, validate and process structured data worldwide.

Using Arelle for ESRS

- Arelle currently supports validation of the draft ESRS XBRL taxonomy
  - Base specification only
  - ESRS specific validations (business rules, etc) will be added to Arelle and open sourced once they are made available.

- Arelle currently supports multi-instance and multi- target functionality

- The Arelle Viewer can display ESRS tagged documents

Let's take a closer look at how Arelle supports ESRS

Multi-Instance / Target Processing Use Cases

- EDGAR Fee Exhbits, SBSEF (multi-instance, private/public)
  - Multi-instance situations (Fee Exhibit, SBSEF)
  - Redacted content or confidential treatment of some instances

- ESEF + ESG (possibly multi-target or multi-instance, all-public)
  - ESG may be comingled with financial data (where IFRS and ESG compatible)
  - ESG may be multi-targeted facts co-habiting financial filing
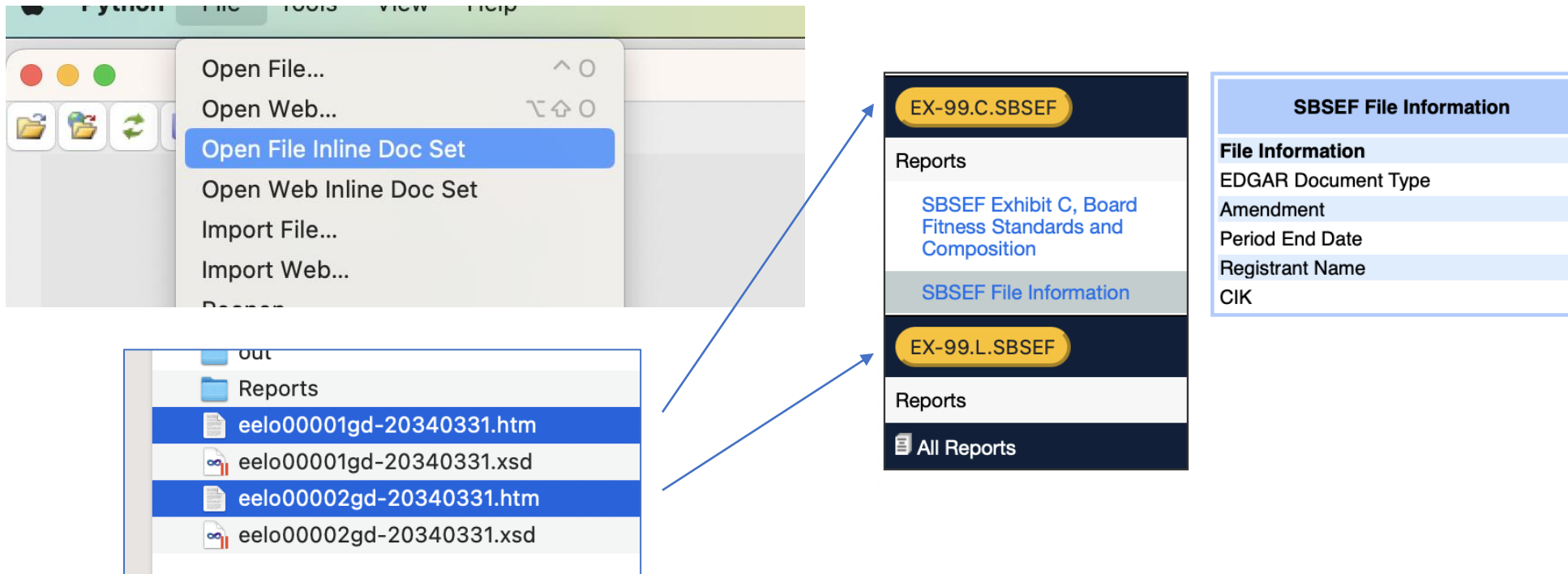 - ESG may be independent instance document

Scenarios of Operation

- ## Multi-instance situations:
  - EDGAR: inline IXDS or xBRL-XML, flat files (in SEC submission SGML wrapper)
    - SBSEF: primary non-XBRL + up to 15 supplemental exhibit IXBRL instances
    - Fee Exhibit: primary IXDS + fee exhibit IXBRL doc
    - SDR: primary xBRL-XML + supplemental xBRL-XML's
  - ESEF: report package of multiple inline report w/ multiple targets each
    - multiple IXDSes with multiple targets (IFRS + DK_GAAP or UK_SEF)
    - ESG: multiple IXDSes with IFRS + multiple ESG targets

- ## EDGAR:
  - receipt: java wrapper allocates doc files to IXDSes
  - GUI (for EDGAR filer): plugin allocates docs to IXDSes

- ## ESEF/ESG:
  - plugin would allocate report pkg report targets to IXDSes

Command Line (as EDGAR uses it)

- Bare bones, leaving out all EDGAR header parameters
  - SBSEF: --plugin EdgarRenderer --disclosureSystem efm-pragmatic -f '[{"file":"…/ sbsef03exc-20231231.htm ","attachmentDocumentType":"EX-99.D.SBSEF "}, {"file":"…/ sbsef03exd-20231231.htm","attachmentDocumentType":"EX-99.D.SBSEF "}, …]' …

  - Fee exhibit: --plugin EdgarRenderer --disclosureSystem efm-pragmatic -f '[{"ixds":[{"file…/F-1-two-instances/pri-doc.htm"}, {"file…/F-1-two-instances/pri-doc-part-2.htm"}],"submissionType":"F-1","attachmentDocumentType":"F-1"}, {"file":"…/F-1-two-instances/EXFILINGFEES.htm","submissionType":"F-1","attachmentDocumentType":"EX-FILING FEES"}]' …

Graphical User Interface in Arelle

- Plugin sorts out files and associates them to IXDSes based on their content

Flat File Partitioning to IXDS

- For validate/EFM

```python
supplementalAttachmentDocumentTypesPattern = re.compile(r"EX-FILING FEES.*|EX-99\.[C-S]\.SBSEF.*")

def isolateSeparateIXDSes(modelXbrl, *args, **kwargs):
    separateIXDSes = defaultdict(list)
    for htmlElt in modelXbrl.ixdsHtmlElements:
        tp = "" # attachment document type inferred from document type and ffd:SubmissnTp
        for qn in ("dei:DocumentType", "ffd:FeeExhibitTp"):
            for elt in htmlElt.iterfind(f".//{{{htmlElt.modelDocument.ixNS}}}nonNumeric[@name='{qn}']"):
                tp = elt.stringValue.strip()
                if tp:
                    break
        separateIXDSes[tp if supplementalAttachmentDocumentTypesPattern.match(tp) else ""].append(htmlElt)
    return [htmlElts for tp,htmlElts in sorted(separateIXDSes.items(), key=lambda i:i[0])]
```

Private / Public Multi-instance Partitioning

- Private and disseminated public instances rendered separately

Pre-partitioned ISDSes (Known in Advance)

- Description at head of inlineXbrlDocumentSet.py plugin

  If there are non-default target documents, the target document identifier can be specified by ixdsTarget

  --file '[{"ixds":[{"file":file1},{"file":file2}...],"ixdsTarget":"xyz"}]'

  If the file source is a zip, CmdLine will discover the inline files in the zip as thus:
  --file '[{"ixds":[{"file":file1.zip}]}]'

  If the file source is a local directory, CmdLine will discover the inline files in the directory as thus:
  --file '[{"ixds":[{"file":dir1}]}]'
  arelleCmdLine --plugin inlineXbrlDocumentSet
  --file '[{"ixds":[{"file":".../PASS-multiple-input-multiple-output-ID1.html"},
  {"file":".../PASS-multiple-input-multiple-output-ID2.html"}],
  "ixdsTarget":"(default)"}]'
  --saveInstance

Workflow with Pre-partitioned ISDSes

- Validate/EFM and EdgarRenderer set up Reports objects

- Edgar Renderer processes each Report (IXDS) in turn

- Edgar Renderer outputs FilingSummary.xml and MetaLinks.json for previously-mustard menu and SEC ixviewer

Workflow without Pre-partitioned ISDSes

- inlineXbrlDocumentSet.py calls plugin IsolateSeparateIXDSes
  - Returns dict of lists of ixHtmlElt trees by doc type

- IXDSes processed as Reports for validate/EFM and EdgarRenderer
  - Compiled into MetaLinks.json and MetaLinks.json by EdgarRenderer

Arelle is ready for ESRS!

- Please visit **arelle.org** for more information about Arelle, to learn how you can contribute to the project or to download the software for free.

Thank you for your attention!

Tom Wacha

Herm Fischer

support@arelle.org