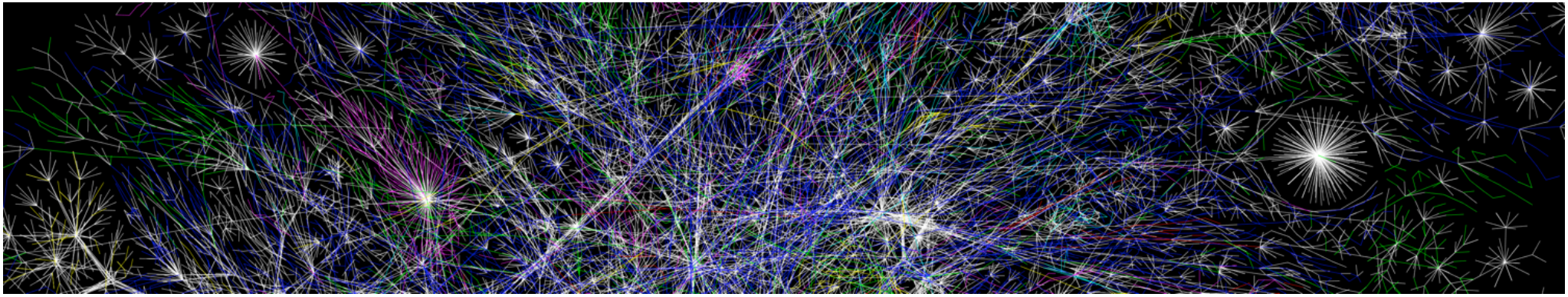
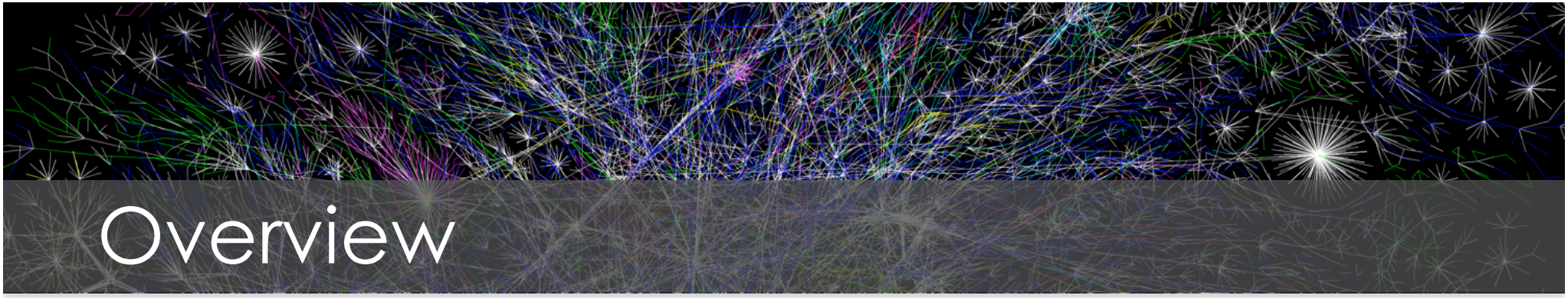


The XBRL Open Information Model

Paul Warren

Herm Fischer

XBRL™
THE BUSINESS REPORTING STANDARD



Overview

- What is the OIM and why does it exist?
- What is xBRL-JSON?
- Key design decisions in xBRL-JSON
- What is xBRL-CSV?
- Practical experiences with xBRL-CSV

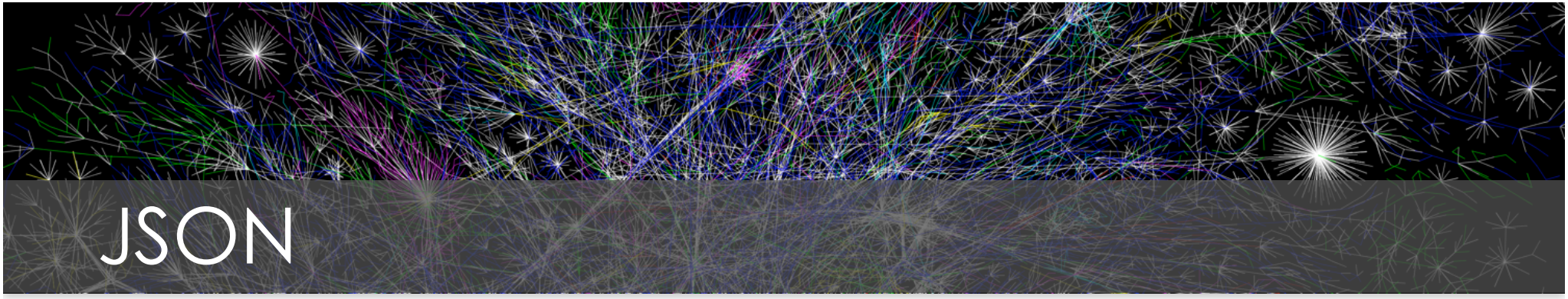
How XML is viewed today

"I'm hoping this will make our unavoidable XML interactions slightly less painful."

- author of xmlpath for Go, in the public announcement of the library

XBRL & XML

< XBRL is wedded to a technology that the average techie doesn't like />



{ People who want to work with data quickly want their data in JSON }

Open Information Model goals

- Break the tight ties between XBRL and its XML syntax
- Achieve simplification: XBRL has some very complex, but little-used features
- Embrace JSON, CSV

OIM specification components

- **OIM** – a syntax independent report model (or "XBRL infoset")
- **xBRL-XML** – mappings between the existing XML and the OIM
- **xBRL-JSON** – mappings between the OIM and a new JSON representation
- **xBRL-CSV** – mappings between OIM and a standard CSV represenation

xBRL-JSON: design priorities

Goal:

Make XBRL data easy to work with

Therefore, prioritise:

1. Ease of consumption
2. Simplicity of representation (readability)
3. Efficiency of consumption
4. Size of representation

Sample JSON fact

```
{
 "id": "f923",
 "value": "1234",
 "numericValue": 1234,
 "baseType": "xsd:decimal",
 "oim:concept": "tax:NumericConcept",
 "oim:entity": "cid:123456789",
 "oim:period": {
     "start": "2015-01-01T00:00:00",
     "end": "2016-01-01T00:00:00"
 },
 "oim:unit": "iso4217:GBP”
}
```

Clark Notation, QNames & SQNames

```
"oim:entity":  "cid:123456789",
```

- QNames in XML are a pain
  - Break simple string comparison (multiple prefixes for a single namespaces)
  - Prefixes can be rebound within a single document (multiple namespaces for a single prefix)
  - People often forget that it's "just a prefix"

Clark Notation, QNames & SQNames

```
"oim:entity":  "cid:123456789",
```

- JSON does not have a native QName mechanism
- We considered Clark notation:

  {http://www.example.com/mynamespace}MyConcept
- Improves ease of consumption, but has a huge impact on readability and size of representation

Clark Notation, QNames & SQNames

```
"oim:entity":  "cid:123456789",
```

- Settled on SQNames:
  - Remove constraint on local part being an NMToken (can start with a number)
  - Require 1:1 mapping between prefixes and namespaces within a document

(latter point supported by a notable xml-dev post dating back to 2002)

Period handling

```
"oim:period": {
    "start": "2015-01-01T00:00:00",
    "end": "2016-01-01T00:00:00"
},
```

- Dates in XBRL contexts have been an endless source of error and confusion:
  - Time component may be omitted
  - If it is, the meaning depends on which element it appears in, in order to support natural, date inclusive representation

Period handling

<startDate>2016-01-01T00:00:00</startDate>
<startDate>2016-01-01</startDate>
both mean "start of 1st Jan"

<endDate>2017-01-01T00:00:00</endDate>
means "start of 1st Jan" aka "end of 31st Dec"
But:
<endDate>2017-01-01</endDate>
means "end of 1st Jan" aka "start of 2nd Jan"

Period handling

Allows you to write:

```
<startDate>2016-01-01</startDate>
<endDate>2016-12-31</endDate>
```

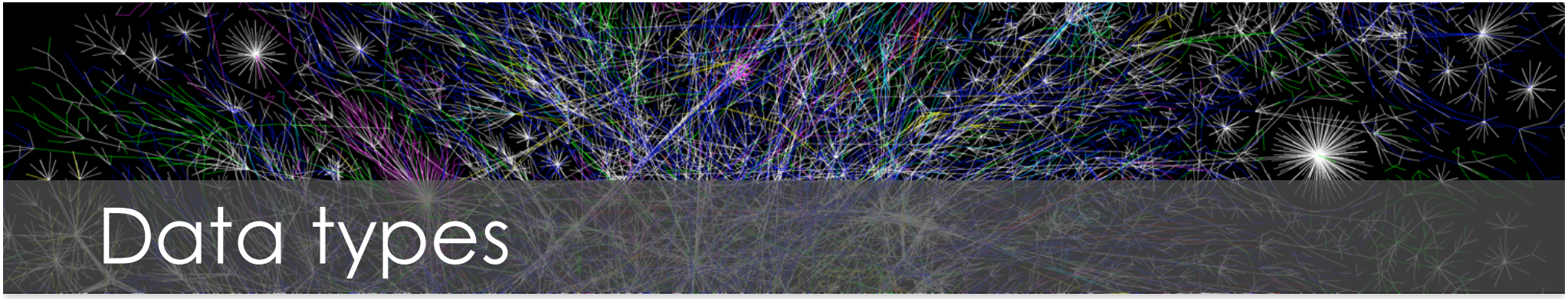
But has caused untold confusion in the process.

Period handling in xBRL-JSON

```
"oim:period": {
    "start": "2015-01-01T00:00:00",
    "end": "2016-01-01T00:00:00"
},
```

- Time component mandatory
- Instants represented as duration where start == end
- PWD uses ISO duration notation:

  `"oim:period": "2015-01-01T00:00:00/2016-01-01T00:00:00"`
- But software support is very poor, so reverted to separate components



Data types

- OIM uses a subset of the XML Schema type system
- Data type information is included in the report as it is necessary to correctly interpret values (particularly QNames)

Data types in xBRL-JSON

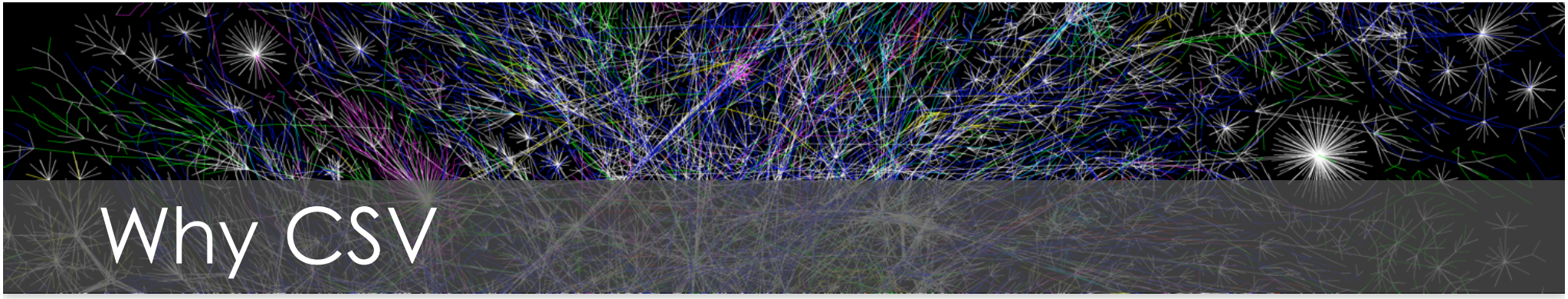
```
"value": "1234",
"numericValue": 1234,
"baseType": "xsd:decimal",
```

- number type in JSON is expected to be double precision
- XBRL numerics effectively require arbitrary precision
- xBRL-JSON provides numericValue for ease of consuming in the “simple case” and “value” as an accurate string representation of the value.

The Fact ID attribute

```
"id": "f923",
```

- IDs (e.g. context and unit IDs) are excluded from the model, as they're irrelevant syntactic detail
- Fact IDs are retained as a mechanism for tracking facts through different representations

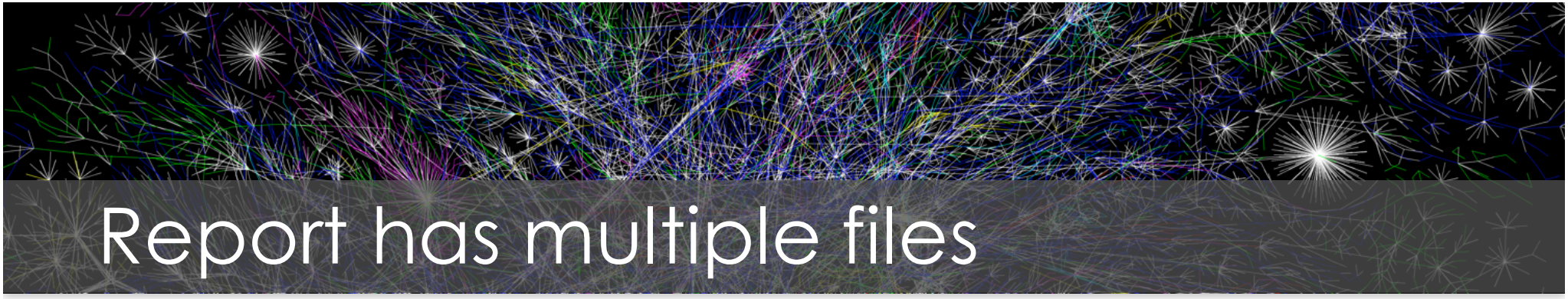


Why CSV

- Widespread long use standard
  - 1972 IBM Fortran and OS/360
  - 2005 IETF RFC4180
  - 2015 W3C Tabular Metadata
- Low overhead
- Accommodates huge data
- Inherently streamable

xBRL-CSV

- Serialize OIM model in CSV
- Full support of OIM features
- Leverage W3C REC for metadata
  - Multiple file reports



Report has multiple files

- {report}-dtsReferences.csv
- {report}-facts{-suffix}.csv
- {report}-defaults.csv
- {report}-prefixes.csv
- {report}-footnotes.csv
- {report}-metadata.json

CSV instance facts

```
"id","stringValue","baseType,"oim:concept","oim:entity",
   "oim:periodStart", "oim:periodEnd"⏎

"f923", "This is a fact value", "xsd:string", "tax:StringConcept", "cid:123456789",
   "2015-01-01T00:00:00","2016-01-01T00:00:00"⏎
```

Multiple fact files

- May organize by table grouping
  - Separate file per table or filing indicator
  - Separate files by common aspects

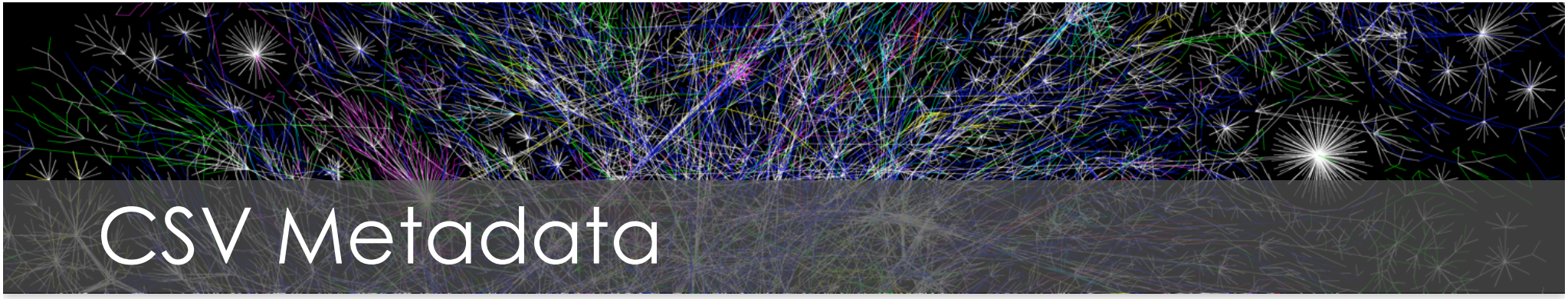
Multiple files and aspect defaults

- Identify fact CSV table files
- Aspect defaults where common to facts of a CSV file
- Support grouping

"file","oim:periodStart","oim:periodEnd","mydim:country"⏎

,"2015-01-01T00:00:00","2016-01-01T00:00:00",⏎

"abc-facts-sales.csv",,,"mydim:southAmerica"⏎

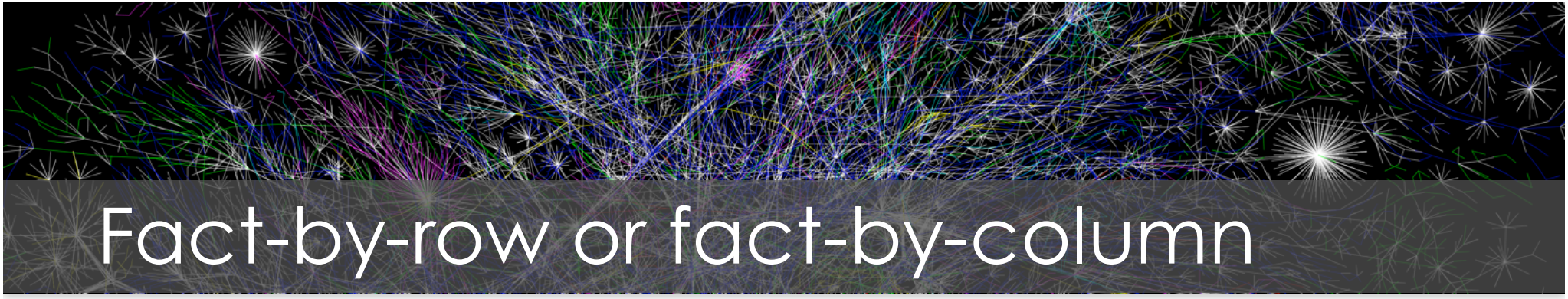


CSV Metadata

- W3C REC: "Model for Tabular Data and Metadata on the Web"
- Model and serialization
- Column types, validation, format
- Column labels
- XML data types and validation

Metadata example

```
{"@context": ["http://www.w3.org/ns/csvw", {"@base": "./"}],
 "tables": [
   {"url": "abc-facts.csv",
    "tableSchema": {
      "columns": [
        {"name": "id", "datatype":  "Name"},
        {"name": "baseType",  "datatype":  "QName"},
        {"name": "stringValue", "datatype":  "string"},
        {"name": "decimalValue", "datatype":  "decimal"},
        ...
        {"name": "accuracy", "datatype":  "float"},
        {"name": "oim:concept","datatype":  "QName"},
        {"name": "oim:periodStart", "datatype":  "dateTime"},
        {"name": "oim:periodEnd", "datatype":  "dateTime"},
        {"name": "oim:unitNumerators", "datatype":  {"base": "QName","separator":  " "}},
        {"name": "oim:unitDenominators", "datatype":  {"base": "QName","separator":  " "}}] } },
   {"url": "abc-prefixes.csv",
    "tableSchema": {
      "columns": [
        {"name": "prefix", "datatype":  "string"},
        {"name": "URI", "datatype":  "anyURI"}] } }
   ...
   ]
}
```



Fact-by-row or fact-by-column

- Flat fact space (basic xBRL-CSV)
  - Like XBRL, one row per fact
  - Factor out common aspects by file
- Table modeled columns (extension)
  - Row-grouped facts in multiple columns
  - Factor facts with common aspects by row

CSV Table-modeled (extension)

```
year,  property,  equipment
2015,  1000,      2000
2014,  3000,      4000
```

- Proposes that the metadata would map:
  - Year column to contribute year to a row's period dates
- Each property and equipment column
  - contribute a fact value,
  - concept aspect for the column,
  - assembles period-start, -end aspects from year column

Discussions of table approach

- Is standardizing tables important?
  - Allow each user to specify own tables
  - Is interoperability important?
- Which table model(s) to leverage
  - W3C Tabular Metadata
  - XBRL Table Linkbase

CSV table use case examples

- SEC rendering output "R files"
  - Could be tabular or columnar CSV files
  - Flat facts space or facts by columns
  - Would that encourage XBRL consumption
- DataAct Gov't Spending output
  - Provide tabular organization to user query