Standards. . .

V3: Compatibility

Challenges for integration and conversion

Data format conformance vs processor conformance

Some basic compatibility relationships

Compatibility backwards and forwards

P-compatibility

Compatibility issues

We discussed some of the challenges involved in integrating different formats and migration

Here we specifically look at the formalities of data standards compatibility.
Especially with respect to adjusting to changing standards

These issues can be can be huge questions for an organization or project, conversions can:
be expensive,
lose information,
break commercial or locally developed applications, tools and workflows,
and complicate relationships with collaborators, suppliers, and customers

At the same time falling behind can
reduce functionality and access to new applications and tools
and complicate relationships with collaborators, suppliers, and customers

Tools for conversion may be available of course, but they may not be reliable, and
obviously any local extensions or adaptations will be particularly vulnerable.

Data format conformance vs processor conformance

Data standards can define conformance for both (i) data and (ii) processing

Data set conformance might require such things as

- a particular character encoding
- particular delimiters
- serialization matching a particular formal grammar
- constraints such as referential integrity, data types
- inclusion of relevant metadata

Processor conformance might require the processing software to

- correctly tokenize
- verify that statements match a particular grammar
- perform additional validation (referential integrity, data types, etc)
- confirm required metadata
- process data sets correctly,
  - e.g., performing particular actions, such as generating a normalized parse tree,
  - or performing calculations, rendering visualizations, etc.
  - displaying an error when processing non-conformant datasets
    - [the processor may also be required to halt, or it may be allowed to continue

Processing conformance may be tested by a specified "test suite" of data sets
Data set conformance is tested by validating softward.

Some basic compatibility relationships

Where **S1** and **S2** are versions of a standard (or schema) the usual sorts of class relationships may be defined:

A dataset *d* is **S1**-*valid* =df *d* conforms to **S1**.

S1 *includes* **S2** =df all **S2**-valid datasets are **S1**-valid

S1 is *equivalent** to **S2** =df **S1** includes **S2** and **S2** includes **S1**

S1 and **S2** *overlap* =df some **S1**-valid datasets are **S2**-valid
[equivalent to: some **S2**-valid datasets are **S1**-valid]

S1 and **S2** *properly overlap* =df some **S1**-valid datasets are **S2**-valid,
and some **S1**-valid datasets are not **S2**-valid,

S1 excludes **S2** =df no dataset is both **S1**-valid and **S2**-valid

NB: ***includes*** **is** *transitive*
overlap and **proper overlap** are not transitive

*whether or not equivalence implies identity is discussed further on next slide

Compatibility backwards and forwards

Consider a standard or schema **S1** that is replaced by a new version, **S2**.

S2 is *backwards compatible* with **S1** =df **S2** includes **S1**

Usually not hard to achieve, usually an objective. i.e. all **S1**-valid datasets are **S2**-valid

S1 is *forwards compatible* with **S2** =df **S1** includes **S2**

Sometimes hard to achieve, so often not an objective. i.e. all **S2**-valid datasets are **S1**-valid

S1 is *equivalent to* **S2** =df **S1** includes **S2** and **S2** includes **S1**

Trivial (?)* i.e. **S2**-validity and **S1**-validity are coextensive

S1 is *p-compatible*** with **S2** =df **S1** properly overlaps with **S2**

sometimes achievable, a very good thing if possible i.e. some datasets are both both **S2**-valid and **S1**-valid.
but some are not

NB *backwards* and *forwards* compatibility are **transitive**, p-compatibility is **not.**

*Actually this depends on what level of representation is intended for standard/schema identity. At the highest conceptual level **S1** and **S2** are the same standard if they determine the same set of conformant datasets, regardless of *how* they determine that set, and so two equivalent standards/schemas are the "same" schema. But here is a practical quesiton: do two schemas that are *syntactically* different (and perhaps in different schema definition languages) define as conformant the same data sets. This is not at all trivial. So if syntactical difference counts as a distinguishing schemas the equivalence is an important notion.

**Not a term in common use.

P-compatibility

In the case of *p-compatibility* there is a non-null intersection of **S1**-valid and **S2**-valid
but not all **S1**-valid datasets are **S2**-valid.

what are the consequences of getting into, or staying in, that intersection?

can existing **S1**-valid files be converted to that intersection?

without loss of information or functionality? [perhaps some can and some can't]

Here is a simple example.

A tag minimization is allowed in **SGML** but not supported in **XML**
In this case conversion is trivial

Not so simple

But XML also does not allow schemas to depart for standard formal grammars
by allowing inclusion or exclusion of elements in element tree.

It is quite a bit more work to bring schemas into line here.

P-compatibility: An example

From 1999-2002 Renear was chair of the Open eBook Publication sWorking Group, which developed OEBPS, an early version ePUB eBook content format.

OEBPS A.2 replaced OEBPS A.1 backwards compatibiltiy was not possible. So A.2 was designed so that large subset of actual and possible A.1 documents would also be A.2 conformant.

This allowed:

1) Many already existing A.1 documents to also be A.2 conformant.
2) Content producers in A.1 shops to continue to use existing software tools and workflows to produce documents that were also A.2 conformant
3) Content producers in A.2 shops to to use new software tools and workflows to produce documents that were also A.1 conformant as well as A.2 conformant.
4) A.1 applications (e.g. eBook production tools and consumer readers) to continue to process A.2 conformant data — if it was in the A.1-compatible subset
5) A.2 applications (e.g. new eBook production tools and consumer readers) to process A.1 conformant data — if it was in the A.1-compatible subset

You can see why *p-compatibility* is a good compromise, if you can’t get it backwards compatibility.

Details changed to protect the innocent.