Standards. . .

V1. Standards and standards organizations

Why standards?

The standards landscape

Standards from a process pov

Why standards?

Very generally:

standards promote reliable efficient communication,
with others now, and with ourselves in the future.

Less generally:

Much data curation involves representing information, if standards are used at every level of the abstraction stack efficient reliable representation is supported.

Much of data curation involves documentation; if metadata standards are used for documentation, then, again, efficient reliable representation is supported.

[Even more specifically: standards support the use of future and third party applications and tools, greatly improving efficiency and reliability and broadening access to users, now and in the future.]

Or with respect to data curation actions:

standards promote validation, authentication, preservation, regulatory compliance, . . . etc. etc.

Standards landscape in US

You know the joke: "The problem with standards is . . ."

NIST 1996 report:

- 93,000 standards produced
- Nearly 700 standard-creating organizations
- US government was the largest single creator and user of standards

Toth, R. (1996). *Standards Activities of Organizations in the United States.* NIST Special Publication 806.

Process issues: Development

Some standards emerge simply as a common practice, without any intention to create a standard for common use.

Some are developed by focused working groups or committees (e.g., ePUB, TEI, DCMI).

Some are developed by larger institutions or agencies (IETF, W3C).

Some developed (or managed) by standards organizations *per se* (e.g. ISO, ANSI, IETF) and designed for wider use.

Many are developed by governments, or global agencies on behalf of multiple governments,

Some standard begin in industry groups, but then are adopted by larger standards bodies to ensure maintenance and impartial evolution.

Process issues: compliance, transparency, access

Compliance may be entirely optional, or required by organizations or governments in certain circumstances.

Usually the requirement is of this sort: if you wish to . . . then you must . . .

i.e., receive our endorsement, use our software, bid on our RFP,
use interstate networks in X nation, sell commercial aircraft in X nation . . .

Most standards development is fairly open to broad participation -- in order to avoid the accusations of partiality that can seriously impeded adoption.

The principle obstacle to participation is the time and experience needed to master complex material, and, in most cases, fairly considerable travel expenses.

Some important agencies making data-related standards

American National Standard Institute
https://www.ansi.org/

ISO
https://www.iso.org/about-us.html

ISO 19115 Geographic information

ISO 14721 Open Archival Information System

ISO 16363 Trustworthy Data Repository

W3C
https://www.w3.org/

Dublin Core Metadata Initiative
http://dublincore.org/

Please spend at least half an hour exploring each site and looking at standards relevant to your work.

V2: Some standard standards maneuvers

Standards are built on standards which are built on standards which are . . .

There are standards, and there are meta-standards (and for more reasons than one)

Being non-standard can be standardized too

Standards are built on standards . . .

Take a close look at almost any standard: XML, ePUB, JSON, etc.

You will see that it is based on other standards

A standard markup language doesn’t need to define its own schema language,

It can specify XML Schema,

and XML Schema in turn doesn’t need to define its own character encoding,

it can specify Unicode,

and so on,

[with great complexity of breadth and depth, especially if “subsetting” takes place
i.e. only portions of the subordinate standards are allowed.
this can make production and validation complicated,
but simplify things for downstream applications]

Meta-standards . . .

Imagine trying to get everyone to agree on a document markup language

Shall it be LaTeX? Or Scribe? Or troff –ms? Or a new one?

Right. It won't be easy. And may not be possible. And you could get hurt trying.

So how about standardize not markup languages, but on *how we define* markup languages.

i.e. how we indicate what our tags are, what patterns or use are legitimate, etc.
[And do it with a processable formal grammar so schemas can be validated and configure applications.]

Then we don't have to agree on so much up front.

We can still keep *trying* to standardize the language itself, and we may succeed (e.g. TEI, JATS, etc.).

But at least we get something important, very important, and necessary in any case, by going meta.

[And that, of course, is why we have standards like XML.]

Standardizing being non-standard . . .

Standards may be extended in standard ways

e.g. the XML internal subset lets elements and attributes be added to a schema

Completely non-standard constructs may be used

For instance, one can use an escaped segment to have the processor skip validation

Departures from standards can be graceful

Don’t just start using a construct because you want to and *your* apps can deal with it.

(e.g. the infamous <blink> tag that was not in the HTML standard).

Instead:

1) add it to the schema so at least it won’t break validation

2) document it so everyone knows what it means or does

Standards wishing to support such departures . . . extensions . . . can specify

That 1) and 2) above are required for conformance

A “fallback”*:

Any *non-standard* construct must be mapped to a *standard* construct that approximates the meaning of the non-standard element or is an appropriate alternative action.

*An approach used in the ePUB standard (International Digital Publishing Federation)

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.