Identity and Identifiers

Contents

V4: An Example: Canonicalization

Here we give an example of *canonicalization*, a central strategy for determining identity of representation or propositional content.

Canonicalization

Canonicalization is a technique for determining representational identity

and is a reasonable proxy for propositional identity.

It is most directly applicable when the representation syntax of the two data sets is the same, but there are

-- variations in encoding

-- alternative synonymous variation within a single language

XML as an example

Imagine two XML files.

Assume they both define the same data structure
(the same labelled graph with attribute/value pairs).

But they could have

- different pretty printing conventions
  - end tags indented or not
  - spaces vs tabs
- Or different character encodings
  - [or different line end encodings (0x0A, 0x0D, 0x0D+0x0A. . .)]
- Or they may have multiple ways to define the same data structure, e.g,:
  - global defaults for attribute values vs local specification
  - random order for attribute/value pairs on element vs . . . Some other random order
  - (and so on).

These quite different files can specify the exact some data structure, but how can we tell?

- You can't use a checksum to determine whether they are logically equivalent
  - A checksum only determines bit, byte, or (sometimes) character "fixity"

This is the problem canonicalization goes after.

A canonicalization recipe

Here is a simple example of how XML canonicalization might work:

To determine whether or not two XML files determine the same data structure.

1. Convert to a single character encoding and normalize line ends.
2. Remove all comments, tabs, non-significant spaces, etc.
3. Propagate all attribute defaults indicated in the schema to the elements themselves
4. Put attribute/value pairs on elements in alpha order
5. Expand all character references
6. Remove any internal schema or declarations.
7. Now test to see if character sequences are identical.

Canonicalized normal forms as proxies for propositional content

[Ok, I see you have noticed we seem to have walked away from identifying propositional content]

Are we giving up?

No, but discretion is the better part of valor, so we are compromising:

When a single data description language is being used a fixity checksum on files in canonical normal form is for all practical purposes our best shot at sameness of propositional content.

Alternative approaches can be imagined, but they are difficult to implement and it isn't clear the results will be much better.

Readings

Required:

Persistent Identifiers, Fixity and Checksums, in *The Digital Preservation Handbook*, Digital Preservation Coalition, 2017.

On the utility of identification schemes for digital earth science data: An assessment and recommendations. Duerr, R. E., Downs, R. R., Tilmes, C., Barkstrom, B., Lenhardt, W. C., Glassy, J., Bermudez, L. E., & Slaughter, P. (2011). *Earth Science Informatics*.

=> Clifford A. Lynch, "Canonicalization: A Fundamental Tool To Facilitate Preservation and Management of Digital Information," *D-lib Magazine*, 5:9 (September 1999).

=> *Canonical XML*. Version 1.0. W3C Recommendation, John Boyer, March 2001. Latest version: http://www.w3.org/TR/xml-c14n.