Digital Object Lineage in Independent Information Systems

Stephan Heuscher

Dynamic and Distributed Information Systems,
IFI, University of Zurich

CSG Seminar for Doctoral Students,
23. September 2008
# Table of Contents

1. **Introduction**
   - Basics
   - Motivation

2. **Main Talk**
   - Digital Object Identification and Lineage
   - Metadata Adhesion
   - Embedding Lineage
   - Implementation

3. **Roundup**
   - Conclusion
Terms and Definitions

Digital Object: An empty or finite sequence of bits.

\[ DO \in \{0, 1\}^* \]  \hspace{1cm} (1)

Digital Object Equality: Digital objects are equal, if all their bits are the same.

\[(|DO| = |DO'|) \land (DO[i] = DO'[i] \forall i) \rightarrow DO = DO'\]  \hspace{1cm} (2)

Information System: Store for digital objects. Digital objects can be retrieved as a whole.

Digital Object Lineage: Directed acyclic graph containing the predecessors of a digital object.

Metadata Adhesion: A qualitative measure for the effort needed to dissociate metadata from the digital object.
How can we answer:

"Which digital objects were used to create the digital object at hand?"

A generic answer to the question will help in:

- Identification of digital objects
- Identity of digital objects
- Attribution and citation of digital objects
  - Academia
  - Digital Rights Management
  - Digital long term archiving
- Repeatability of in silico experiments
- Lessen dependence on information systems (identifier resolution) for digital objects
The Dilemma
Does the identity of the digital object depend on an information system?
Identity is the basis for lineage.

Identity is defined on the basis of equality.

Identifiers identify digital objects.

Uniform Resource Identifiers (URI)

- Depend on information systems (resolution services) to retrieve digital objects.
- Do not explicitly reference stable digital objects.

Intrinsic Properties

- Only depend on the content of the digital object.
Digital Object Identifier: A digital object identifier is result of an identification function $ID$ on the digital object that must ensure that equal digital objects share the same result.

$$DO = DO' \rightarrow ID(DO) = ID(DO') \quad (3)$$

**ID functions**

- Functions extracting parts of the digital object (e.g., header fields).
- We use cryptographic hash functions on the content of the digital object since they provide identification as well as integrity (collision resistance).
Do we need a new protocol?

- To resolve the identifier, resolvers are needed, however
- Identifiers can be embedded in existing identifier systems and uniquely identify digital objects. Examples:
  - U.S. Dept. of Health and Human Services, National Center for Health Statistics. *National Hospital Discharge Survey, 1997*[Computer file]
    - md5:689d07f9281298a08e97bb095167fa84
  - [http://example.org/do.pdf?md5=a3cca2b2a...](http://example.org/do.pdf?md5=a3cca2b2a...)
- The recipient (or his software) of the digital object can evaluate the integrity of the received object.
Lineage Metadata

- Builds on the information system independent identification of digital objects.
- Captures the parent-to-sibling relationship between digital objects.
- Starting point for further investigations on the digital objects involved.
- Needs to be recorded by the systems managing the digital object.

Future Work
  - Integration of processing metadata.
Lineage Example
Example Workflow in an Stock Photography Agency

Introduction
Main Talk
Roundup
Metadata Adhesion

Referencing
Reference Types

Metadata references digital objects

- *Explicit* references are unambiguous references to the digital object.
- *Implicit* references rely on (system-)immanent references to the digital object.
- *Hard* references feature actively enforced integrity constraints.
- *Soft* references feature no integrity constraints nor active constraint enforcement. Soft references may be *dangling*. 
## Metadata Adhesion

### Examples for Reference Types

<table>
<thead>
<tr>
<th>Reference Types</th>
<th>Explicit</th>
<th>Implicit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard</td>
<td>Foreign to primary key relationship in a database management system. File system metadata for a file.</td>
<td>- (We have not found any examples for implicit hard references.)</td>
</tr>
</tbody>
</table>
The Dilemma
Does the identity of the digital object depend on an information system?
**Metadata Adhesion**

**Packaging**
Merging the digital object and the metadata

**Encapsulation:** Generating a package which contains the metadata and the digital object in a suitable form.

\[
\text{Package} = \text{enc}_1(MD, \text{enc}_2(DO))
\]  

(4)

**Format-specific Embedding:** Integrating the metadata with the digital object so that the metadata becomes part of the digital object. The digital object retains its format and becomes a package.

\[
\text{Package} = \text{embed}_{\text{format}}(DO, MD)
\]  

(5)

**Binary Append:**Appending the metadata format specifically to the digital object. The digital object retains its format and becomes a package.

\[
\text{Package} = DO\|\text{enc}_{\text{format}}(MD)
\]  

(6)
Embedding Metadata in Digital Objects

Basic Requirements

- High metadata adhesion
- Identification of the metadata without knowledge of the digital object format
- Easily accessible metadata
- Accomodate arbitrary metadata, therefore also lineage metadata
- Verifyable integrity
- Simple, easy to understand
Embedding Lineage

The SIMPLE Method
Simple Identifiable Metadata with Persistent Lineage Embedding

The mandatory SIM part

- Simple: Overarching goal, principle of least astonishment, natural language descriptions
- Identifiable Metadata: Definition of metadata container on the byte level, XML technology, `<metadata describes=’ as start (“magic number”), metadata ends with ‘</metadata>’

The optional PLE part

- Persistent Lineage: Lineage is encoded as XML tree of predecessor digital objects.
- Embedding: Using format-specific embedding or binary append.
Sealing the SIMPLE Digital Object (SDO)
Protection against Unintentional Changes
SIMPLE Prototype
Implementation for Binary Append

- Supports XML, HTML, and PDF as *open ended* formats.
- All other formats are treated as *confined* formats.
- Implementation in Java 1.4 or greater.
- Command line, web, and automatic build tool interfaces.
- Modules of the prototype and their interaction:
Upload

Uploaded files will be altered, so that they contain the metadata provided. They will also be sealed. Anyone can access them. (Back to the files.)

Please specify file and metadata

Description: <textarea>valid XML</textarea>

Language: English  
Hashing: sha1 & md5  
File (maximum size: 1MB): Browse...  
Upload

Uploaded files can not be deleted and are accessible for anyone with internet access.

Links

MD5 Checksum creator: [http://www.fourmilab.ch/md5/](http://www.fourmilab.ch/md5/)  
[http://holger.thoelkinger.name/skripten/md5](http://holger.thoelkinger.name/skripten/md5)  
Hex editor (look at the bytes of a file): [http://www.chmaas.handshake.de/delphi/freeware/xvi32/xvi32.htm](http://www.chmaas.handshake.de/delphi/freeware/xvi32/xvi32.htm)
Identification of digital objects must be decoupled from specific information systems to allow digital object lineage.

Prototypically shown integration of arbitrary metadata with digital objects.

Future Work

- Case study with the state archives of Appenzell Ausserrhoden (under way).
- Integration of processing metadata with the lineage metadata.
Some open questions

- How important is the inclusion of processing metadata in lineage?
- Does the identification need to be part of the digital object?
- Are there privacy or trade secret concerns when capturing lineage?

Thank you for your attention.