Towards Persistent Identification of Resources in Personal Information Management

Stefan Haun and Andreas Nürnberger

Data and Knowledge Engineering Group,
Faculty of Computer Science,
Otto-von-Guericke-University Magdeburg, Germany
http://www.findke.ovgu.de

Abstract. Persistent identification is necessary for recognition, dissemination and (external) cross-references to digital objects. *Uniform Resource Identifiers (URIs)* provide an established scheme for this task, but do not guarantee stable and persistent identification. In the context of (personal) archives, stability is needed when references are be stored on a medium where later changes to identifiers cannot be corrected at all or only with a very large overhead, such as WORM media or tape archives. Additionally, resources like contacts or appointments do not have a URI, while other URIs, such as file system paths or the IMAP URI, are unstable by design and cannot represent the dynamic aspects of Personal Information Management (PIM). This paper discusses problems of archiving that arise with entity identification in PIM, especially on the example of the personal file system.

1 Introduction

During the past decades, our everyday work-life has rapidly moved from the real world to the digital domain. Then we used to put our hands on paper files, books, business cards, calendars or photographs, nowadays everything is a digital object stored on the computer, in the smart phone or the Cloud. Still we need ways for virtually grabbing and referencing those objects.

The *Uniform Resource Identifier* (URI) serves exactly this need: to identify digital resources for later look-up and to reference objects. However, there is only a very generic specification for how a URI must look like. This results in high flexibility and viability for many applications, but URIs may not match the requirements posed by the task. For example, the proposed URI for a file path is unstable in the way that those URIs become invalid whenever the file is moved. Therefore external references in objects no longer point at that file and have become stale and thus useless.

Even worse, if these objects are archived, changes to the stored data are often prohibited either by policy or due to the storage medium. For example, A CD-ROM is read-only and re-writing a tape storage takes much effort. Therefore identifiers need to be stable so that existing archives are not broken.

Having reliable identifiers for digital objects is a requirement to mine and store relationships between those entities, especially in dynamic environments such as Personal Information Management. With an integrated view novel interaction concepts, such as graph based interaction, are possible. These approaches enable the user to explore large graphs–such as the graph emerging from connected objects in Personal Information Management–in order to find specific elements without the need for keyword-based search or to get an overview on structures in the personal information space (e.g. [4]).

The paper starts with a short analysis of related work and a definition followed by an elaboration of requirements and problems with stable identification systems. The findings are further discussed on the example of a personal file system to point out some problems that arise with current systems.

2 Related Work

Entity identification has uses in several areas: In geometry, entity identification is a known problem when objects change, but reference points must be recognized [16]. In the context of the *World Wide Web*, identification refers to digital resources. The W3C[1] creates and maintains standards related to the Internet and especially the WWW, which are highly relevant to this paper. In the *Persisting Identifier Linking Infrastructure (PILIN) project* options for identification of public entities and necessary infrastructures are researched [9]. Semantic annotation of data, for example in linked-data sets, induces the problem of finding feasible identifiers in a specific domain, such as the files [12] or biological data [10]. Analysis of stability and reliability of digital identifiers can be found in the field of *digital forensics*, e.g. a survey of Message-IDs in e-mails [11].

No related work could be found towards building stable identifiers in Personal Information Management and regarding personal archives, hence this paper aims at starting a discussion in this direction by identifying some general problems based on examples of stable URIs and personal file systems.

3 Persistent Identifiers

3.1 Identifiers

A general definition for the term *identifier* can be found in [9]:

> Any association of a name with a thing – by anyone – establishes an identifier. A name is not an identifier unless it identifies something. (E.g. an unassigned phone number is a name, but not an identifier.)

Digital identifiers exist in contexts which are seldom made explicit or included when citing an identifier.

[1] World Wide Web Consortium (W3C) http://www.w3.org/

RFC 3986 [1] specifies the *Uniform Resource Identifier (URI)* as "a compact sequence of characters that identifies an abstract or physical resource". URIs can be divided into *Uniform Resource Names (URN)*, denoting the name of a specific resource, and *Uniform Resource Locators (URL)*, referring to the location of a resource. However, a formal distinction is often neglected based on the findings from RFC 3305 [5]. Throughout this paper, the term *URI* will be used whenever a further distinction is not necessary. Yet it is important to notice that an identifier in terms of a URI, while referencing a resource, not necessarily has to point at this resource in the sense of a URL.

When looking for an identifier, the URI makes a good choice. Sharing the syntax with URLs, any entity—real, abstract or on the Worldwide Web—can be identified without introducing a new standard.

3.2 Requirements

The URI specification defines requirements regarding syntax and semantics of a URI. However, it is agnostic regarding requirements for a system using the URI. This leads to high flexibility, but demands further analysis in the context of an application using the URI. In [7, 13] the following functional requirements for resource identification systems are defined:

Global scope Identifiers are location-independent and have the same meaning everywhere.

Global uniqueness Identifiers are unique, i.e. one identifier does not reference information associated with multiple resources.

Persistence Identifiers uniquely reference resources beyond their lifetime. This specifically means that an identifier must not decay when the resource is no longer available and, in a broader interpretation, that an identifier must not be re-used for a resource once it has been used.

Scalability Identifiers are scalable and can be assigned to any resource.

Legacy support Identifiers can support legacy identification schemes to the extent that these satisfy minimum requirements.

Extensibility Identification schemes can accommodate future extensions.

Independence Responsible authorities maintain and assign resource identifiers within a given system.

Resolution Identifiers are supported by services that enable their translation.

The W3C demands **opacity** for all identifiers [3]:

> *Axiom: Opacity of URIs* The only thing you can use an identifier for is to refer to an object. When you are not dereferencing, you should not look at the contents of the URI string to gain other information.

URI opacity has been a source of many debates, especially since technical solutions like the GET method for submitting HTML form content explicitly violates this rule.

A more ontological approach towards identifier requirements can be found in [9]. However, the discussion goes beyond the scope of this paper.

3.3 Resolution and Retrieval

In order to obtain the resource denoted by an identifier, the identifier must be *resolved* to a locator which allows to *retrieve* the resource [9]. According to [7] there are basically two ways of implementing persistent identifier management systems:

First, identifier schemes based on *Uniform Resource Names (URN)* as defined in RFC 2141 [6]. Each URN contains a globally unique part, denoting the namespace, and a namespace specific string, containing the actual reference. The reference data is a complete set of all attributes necessary to identify the references object. For example, the URN `urn:ISSN:0259-000X` references the journal with ISSN *0259-000X*. With this information, a copy can be obtained from the local library or a respective web service. As the URN does not contain any information about the storage of the referenced entity, changes to the storage paradigm do not render the identifier invalid.

A second option are handle-based systems: Instead of encoding the information needed to find an entity into the identifier, a handle to a database entry in a resolver is created. In order to retrieve the entity, i.e. *resolve* the identifier, a look-up call to the resolver is necessary. The identifier is then transformed in either the resource itself or a different identifier which can be used to retrieve the entity. Among many, a well-known handle-based example is the *DOI® System*, maintained by the *International Digital Object Identifier Foundation (IDF)*[2]. The IDF hands out unique prefixed for publishers, who create unique identifiers in their own namespace. Thus each digital document, such as publications in conference proceedings, can be referenced without knowing its actual storage location. The DOI resolver maps a certain DOI to a URI in the database, which can be used to retrieve the document. In this case, the resolution consists of multiple steps, as the IDF does not store the documents themselves but identifier for the storage locations. Other known handle systems are *Persistent Uniform Location Locators (PURL)* [3] and the *Archival Resource Key*[4]. Please refer to [15] for further information on those systems.

Many systems are handle based and when the task of resource identification arises, handles and central registries seem to be the preferred choice. In the context of Internet connectivity and centralized systems this makes sense: Having a control entity avoids naming clashes and makes it simple to decouple identifier and resource location. However, a handle based system also requires the availability of a handle database. In the case of personal archives, it may be necessary to store the resolution database alongside with the archive. This may be possible for a complete snapshot with stable references, but if the references may still change, the archived database will be outdated and if only parts of the personal information is archived, the resolution database must either be split or very likely results in a large storage overhead. For an offline scenario or when data is only available locally, e.g. on the personal computer, a handle

[2] http://www.doi.org
[3] http://purl.oclc.org/
[4] https://confluence.ucop.edu/display/Curation/ARK

database may not be feasible at all. Here the information needed for resolution must become a part of the identifier itself. There are URI specifications for many identifiers in the PIM context, such as file paths or e-mails, but these often do not take stability into account.

The following case study shows problems that may arise when identifying resources in the personal file system.

4 Case Study: Personal File System

Hierarchical file systems are still the main way of storing information on a personal computer. PIM systems have to provide means of identifying those files in order to create stable references.

Several systems have been devised towards file identifications, all with their own strengths and caveats. A small selection shall be discussed to give an overview on the problem field.

File Path A naive approach is given by the file path, i.e. the location of the file in the local file system. RFC 1738 [2] specifies in Section 3.10 the `file:` URI scheme "to designate files accessible on a particular host computer." The solution has several caveats: First, path names are only *unique* in the scope of the host. If a reference to the file is transferred to another host, the context is lost and it becomes invalid. The uniqueness problem can be solved by adding user and host to the file path. While the URI scheme supports this full qualification of file paths, most computers do not have a globally unique host name, i.e. a name in the scheme `hostname.domain` anymore. The second caveat is much more pressing: Changing the location of a file–which is denoted by the file path–is a common and intended interaction with file objects, i.e. everyday interaction with files will result in broken links. The file URI scheme is not stable in the sense that is becomes invalid as soon as a file is moved or renamed, even though the file still exists. Using the file path to reference a file is only feasible if the file cannot be moved, which is the case for certain system files.[5] Other means of identification are needed to reference files on a computer, than the paths built-in into modern operating systems.

Magnet Links A widespread solution for referencing files based on their content is provided by *magnet links*[6]. Although not listed as a standard, the `magnet:` prefix can be found in the list of URI schemes[7]. Instead of a file path, the URIs are generated from a hash code computed from the binary content of the file, such as SHA-1 [8]. To resolve the URIs, a mapper must keep track of file paths matching the given hash. When a file is moved or renamed, i.e. the file path changes, the mapper database must be updated accordingly. Magnet-link based

[5] For example, the *Filesystem Hierarchy Standard* defines a quite rigid structure for *Unix*-based systems. (http://www.pathname.com/fhs/pub/fhs-2.3.html)

[6] http://magnet-uri.sourceforge.net/

[7] http://www.iana.org/assignments/uri-schemes/uri-schemes.xhtml

URIs are not very common on local file systems for two reasons: First, they have a rather negative reputation, as these identifiers are mostly used in P2P file sharing. Here they serve the purpose very well, as they identify a certain content without knowledge of its location. Second, magnet links become invalid if the file content changes–another intended use of files in Personal Information Management.

Heuristics When both file path and file content change, heuristics must be used to identify a file. The *GIT version control system*[8] tracks changes in the file path by comparing the content of edited files, i.e. if a file is removed and a new file with sufficiently similar content appears, the change is logged as "renamed" instead of "new". Similar, changes in the file path could be detected by monitoring the file system operations, which is supported by common operating systems. Methods from *duplicate detection* can be employed to restore links by providing candidates to replace a broken file reference. Spinellis [14] presents an approach that augments file URIs with data from index vectors, so that broken links can be restored based on previously indexed content, i.e. a search query that will most likely recover the file is attached to the file path. Those approaches cannot guarantee stable references, as the changes cannot be reflected in the generated URI. However, they provide means of recovering a broken URI and represent a step towards stable file links.

Version Control Systems In an assumed system where content cannot be deleted but only updated, such as found in a version control system, the problem boils down to two distinct cases:

1. The generated identifier references a certain version of a file, i.e. a stable content.
2. The generated identifier references a certain path of a file, i.e. *move* or *rename* operations will not be applied.

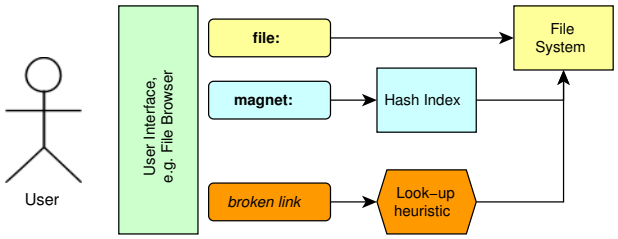

Fig. 1. Schematics of a file resolver combining several approaches

[8] http://git-scm.com/

To create a stable identifier the intended use of a file must be known to select the right approach. Figure 1 shows how the described components could work together to resolve file links. In current operating system this leads to a semantic gap between the available data and the user's intention. It is left to future research to determine if and how this gap can be closed, e.g. based on the file type, by observation of user interaction with the file or by taking the file's provenance into account.

5 Conclusion

In this paper we have discussed some fundamental problems that arise when persistent identifiers for objects from the PIM domain are needed. Requirements and modeling of identifiers cover a growing research field, however, PIM solutions still rely on locally generated identifiers instead of stable, global URIs. In a case study some problems with persistent identifiers have been shown: Stable file links require more attention and may even lead to a semantic gap where the system cannot decide how to generate the correct link.

Future work includes research on remaining PIM entities, specifically e-mails, contacts and appointments as well as further tests on implementation of these identifiers and tests towards reliability in different data sets. The file case study has shown that reliable persistent identification will need methods from machine learning to repair broken links and determine the intended use of an object. Semantic archives may provide the necessary information for re-assigning an object to its previous link if a connection has been lost, e.g. by comparing meta-information. Finally, when persistent identification is possible, user interfaces and semantic desktop applications can make use of those identifiers.

Acknowledgments

Parts of this paper have been researched within the scope of the SENSE project which is funded by the Federal Ministry of Education and Research, German Aerospace Center. It is part of the *KMU-Innovativ: IKT* campaign and goes by the funding numbers FKZ 01IS11025E.

References

1. Berners-Lee, T., Fielding, R., Masinter, L.: Uniform Resource Identifier (URI): Generic Syntax. RFC 3986 (January 2005), http://tools.ietf.org/html/rfc3986
2. Berners-Lee, T., Masinter, L., McCahill, M.: Uniform Resource Locators (URL). RFC 1738 (December 1994), http://tools.ietf.org/html/rfc1738
3. Berners-Lee, T.: Design Issues, chap. Universal Resource Identifiers – Axioms of Web Architecture (1996), http://www.w3.org/DesignIssues/Axioms.html
4. Haun, S., Gossen, T., Nürnberger, A., Kötter, T., Thiel, K., Berthold, M.: On the Integration of Graph Exploration and Data Analysis: The Creative Exploration Toolkit, Lecture Notes in Computer Science, vol. 7250, pp. 301–312. Springer Berlin Heidelberg (2012)

5. Mealling, M., Denenberg, R.: Report from the Joint W3C/IETF URI Planning Interest Group: Uniform Resource Identifiers (URIs), URLs, and Uniform Resource Names (URNs): Clarifications and Recommendations. RFC 3305 (August 2002), http://tools.ietf.org/html/rfc3305
6. Moats, R.: URN Syntax. RFC 2141 (May 1997), http://tools.ietf.org/html/rfc2141
7. Morgan, H.: Persistent Identification of Digital Resources – Environmental Scan. Tech. rep. (2008)
8. National Institute of Standards and Technology, USA: FIPS PUB 180-4: Secure Hash Standard. Federal Information Processing Standards Publication (March 2012), http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf
9. Nicholas, N., Ward, N., Blinco, K.: Abstract Modelling of Digital Identifiers. Ariadne issue 62 (January 2010), http://www.ariadne.ac.uk/issue62/nicholas-et-al
10. Pasquier, C.: Biological data integration using Semantic Web technologies. Biochimie 90(4), 584–594 (April 2008)
11. Pasupatheeswaran, S.: Email 'Message-IDs' helpful for forensic analysis? In: Proceedings of the 6th Australian Digital Forensics Conference. School of Computer and Information Science, Edith Cowan University, Perth, Western Australia (2008)
12. Schandl, B., Popitsch, N.: Lifting File Systems into the Linked Data Cloud with TripFS. In: Bizer, C., Heath, T., Berners-Lee, T., Hausenblas, M. (eds.) LDOW. CEUR Workshop Proceedings, vol. 628. CEUR-WS.org (2010), http://dblp.uni-trier.de/db/conf/www/ldow2010.html#SchandlP10
13. Sollins, K., Masinter, L.: Functional Requirements for Uniform Resource Names. RFC 1737 (December 1994), http://tools.ietf.org/html/rfc1737
14. Spinellis, D.: Index-based persistent document identifiers. Inf. Retr. 8(1), 5–24 (Jan 2005)
15. Tonkin, E.: Persistent Identifiers: Considering the Options. Ariadne issue 56 (July 2008), http://www.ariadne.ac.uk/issue56/tonkin
16. Wang, Y., Nnaji, B.O.: Geometry-based semantic ID for persistent and interoperable reference in feature-based parametric modeling. Comput. Aided Des. 37(10), 1081–1093 (Sep 2005)