Internet-Draft: draft-kunze-ark-00.txt
ARK Identifier Scheme
Expires 22 August 2001

J. Kunze
22 February 2001

The ARK Persistent Identifier Scheme

(http://www.ietf.org/internet-drafts/draft-kunze-ark-00.txt)

Status of this Document

Distribution of this document is unlimited. Please send comments to jak@ckm.ucsf.edu.

1. Abstract

This document introduces the ARK (Archival Resource Key) scheme for persistent naming, describes the ARK syntax, and shows how ARK services work. These services are defined, for now, over the HTTP protocol, given its current reign in networked information retrieval. The first step is to make an ARK actionable by giving it a semantically inert (for identity comparison) hostport combination as a prefix. This requires discovering a Name Mapping Authority (an ARK service provider) based on the Name Assigning Authority that created the ARK. Two methods for discovery are proposed: one file based, the other based on the DNS NAPTR record. The resulting string is then submitted to a web browser as the basis for the three core ARK services underpinning credible claims of persistence: access to objects, to their descriptions, and to the commitments made regarding their preservation. Different services are activated by a set of

conventions for appending a `?' followed by an ARK "request". Thus,

upon release an ARK identifies not one, but three information units (data, metadata, and policy), each unit accessible under the one identifier.

2. Introduction

This document describes a scheme for the high-quality naming of information resources. The scheme, called the Archival Resource Key (ARK), is ideally suited to long-term access and identification for any information resources that accommodate reasonably regular electronic description. This includes digital documents, databases, and software, as well as physical objects (such as books, bones, and bottles) and abstract objects (chemicals, diseases, vocabulary terms, performances). Hereafter the term "object" refers to an information resource. The term ARK itself refers both to the scheme and to any single identifier that conforms to it.

Schemes for persistent identification of network-accessible objects are not new. In the early 1990's, the design of the Uniform Resource Name [URN], responding to the failure rate of URLs in practice, recognized the promise of indirect, non-hostname-based naming and the need for responsible name management. Meanwhile, promoters of the Digital Object Identifier [DOI] succeeded in building a community of providers around a mature software system that supports name management. The Persistent Uniform Resource Locator [PURL] was a third scheme that had the unique advantage of working with unmodified web browsers. The ARK scheme is a new approach.

A founding principle of the ARK is that persistence is purely a matter of service. Persistence is neither inherent in an object nor conferred on it by a particular naming syntax. Rather, persistence is achieved through a provider's successful stewardship of objects and their identifiers. The highest level of persistence will be girded by a provider's efforts to develop contingency, redundancy, and succession strategies. It is further safeguarded to the extent that a provider's mission is shielded from marketplace and political instability.

2.1. Three Reasons to Use ARKs

The first requirement of an ARK is to give users a link from an object to a promise of stewardship for it, as made by an identified service provider. No one can tell if successful stewardship will take place because no one can predict the future. Reasonable conjecture, however, may be based on past performance. There must be a way to tie a promise of persistence to a provider's demonstrated or perceived ability -- its reputation -- in that arena. Provider

reputations would then rise and fall as promises are observed variously to be kept and broken. This is perhaps the best way we have for gauging the strength of any persistence promise.

The second requirement of an ARK is to give users a link from an

object to a description of it; possession of an identifier without a description does not constitute positive identification. Identifiers common today are relatively opaque, though some may contain ad hoc clues pertaining to fleeting lifecyle events, such as entry into a filesystem hierarchy. Possession of both an identifier and an object is some improvement, but positive identification may still be elusive since the object itself need not include a matching identifier or be transparent enough to reveal its identity without significant scrutiny and background research. In either case, what is called for is a record bearing witness to the identifier's association with the object, as supported by a recorded set of object characteristics. This descriptive record can act as a kind of information "receipt" that allows users and archivists briefly to inspect a retrieved object for plausible match with some recorded characteristics, for example, its title and size. (MD5 checksums as identifiers permit an easy computation to verify its association with an object, provided the object remains bit for bit unchanged over time.)

The final requirement of an ARK is to give users a link to the object itself (or to a copy) if at all possible. Persistent access is the central duty of an ARK, with persistent identification playing a vital but supporting role. Object access may not be feasible for various reasons, such as catastrophic loss of the object, or licensing agreements that keep an archive "dark" for a period of years. But attempts to simplify the persistence problem by decoupling access from identification and concentrating first on the latter are of questionable utility. A perfect system for assigning forever unique identifiers might be created, but if it did so without reducing access failure rates, no one would be interested. The central issue -- which may be summed up as the "HTTP 404 Not Found" problem -- would not have been addressed.

2.2. Organizing Support for ARKs

Co-location of persistent access and identification services is natural. Any organization undertaking persistent identification and description is in an advantaged position to undertake persistent access, and vice versa. The former task becomes all the easier if the organization controls, owns, or otherwise has clear access to the objects. Similarly, the latter task (persistent access) cannot be managed without at least internal support for the former task. Thus, organizing ARK services under one roof tends to make sense.

ARK support is not for everybody, as the bar for participation is high. By requiring specific, revealed commitments to preservation, object access, and description, ARK services are not cheap. On the other hand, it would be hard to grant credence to a persistence promise from an organization that could not muster the minimum ARK

services. Still, there is a business model for building an ARK-like, description-only service on top of another organization's full complement of ARK services; for example, there might be competition at the description level for abstracting and indexing scientific literature archived in an open pre-print repository. Such a business

model, however, would benefit from persistence without directly supporting it.

2.3. A New Definition for Identifier

Talking about persistent identifiers can be very confusing because the term "identifier" has not been carefully defined. The best working definition thus far comes as a side effect of defining the Uniform Resource Identifier in [RFC2396].

> Identifier (RFC2396):
> a sequence of characters with a restricted syntax,
> that can act as a reference to something that has identity

The term works for that document, but things break down when it is employed for persistence. Troubling phrases arise, such as

> "we want an identifier that does not break..."

An identifier seems to be a sequence of characters, yet breakage isn't really about them. If it is the reference that breaks, fingers point to a wildly diverse group of suspects: browser manufacturers, the maintainer of the page where the link was found, the syntax of the link itself, the DNS administrator, the firewall, the educational system, etc.

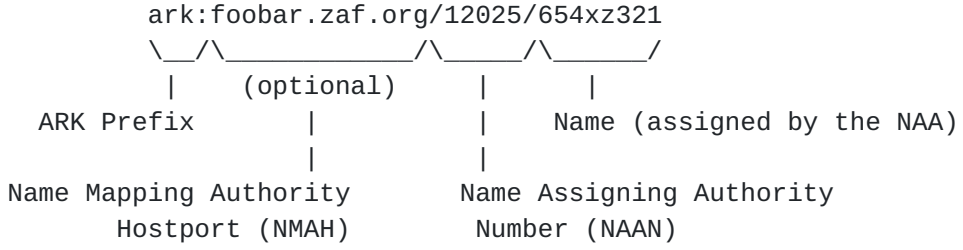
The following new definition is proposed.

> An identifier is an _association_ between a string (sequence of characters) and an information resource. That association is made manifest by a record (e.g., a cataloging record) that binds the identifier string to a set of identifying resource characteristics.

The identifier (association) is now vouched for by a metadata record, and that association is made public in so far as the record is shared; for example, an internal identifier from an organization has limited value to outsiders since the nature of the association has not been disclosed. Without evidence of that association, a sequence of characters may not be recognizable as an identifier. The metadata record can act as a kind of association "receipt" or "declaration". Best of all, it now makes sense to speak of an identifier (an association) that "breaks".

3. ARK Anatomy

An ARK is represented by a sequence of characters (a string) that always begins with the prefix "ark:". Here is a diagrammed example.



3.1. The Name Mapping Authority Hostport (NMAH)

After the prefix may appear an optional Name Mapping Authority Hostport (NMAH) that is a temporary address where ARK service requests may be sent. It consists of an internet hostname or host-port combination having the same format and semantics as the host-port part of a URL. The most important thing about the NMAH is that it is semantically inert from the point of view of object identification. In other words, ARKs that differ only in the optional NMAH part identify the same object. Thus, for example, the following ARKs are equivalent:

```
ark:foobar.zaf.org/12025/654xz321
ark:sneezy.dopey.com/12025/654xz321
ark:/12025/654xz321
```

The NMAH makes it easy to derive an identifier that is actionable in today's web browsers (i.e., a URL). This amounts to replacing the "ark:" prefix with "http://". The first example ARK above thus becomes

http://foobar.zaf.org/12025/654xz321

The NMAH part is temporary, disposable, and replaceable. Over time the NMAH will likely stop working and have to be replaced with a currently active service provider. This relies on one of the NMAH discovery processes outlined in a later section. Meanwhile, a carefully chosen NMAH is as valid as any internet domain name, and may last for a decade or longer. Users should be prepared, however, to replace the NMAH because the one found in an ARK may have stopped working at any time.

The above method for creating an actionable identifier from an ARK (replacing "ark:" with "http://") is also temporary. Assuming that the reign of HTTP in information retrieval will end, one day ARKs will have to be converted into new kinds of actionable identifiers. If ARKs see widespread use, web browsers would presumably evolve to

perform this simple transformation automatically.

3.2. The Name Assigning Authority Number (NAAN)

The next part of the ARK is the Name Assigning Authority Number (NAAN) enclosed in `/' (slash) characters. This part is always required, as it identifies the organization that originally assigned the Name of the object. It is used to discover a currently valid NMAH and it also provides top-level partitioning of the space of all ARKs. NAANs are registered in a manner similar to URN NIDs, but they are pure numbers consisting of 5 digits or 9 digits. The first 100,000 NAAs fit compactly into the 5 digits, and if growth warrants, the next billion fit into the 9 digit form. In either case the fixed odd number of digits leaves clues to reduce confusing them with, say, 4-digit dates.

3.3. The Name Part

The final part of the ARK is the Name assigned by the NAA, and is also required. The Name is a string of printable ASCII characters less than 128 bytes in length, but the characters `/', `.', and `?' are reserved and must not be used. The length restriction leaves room to append ARK requests to an ARK and transport them within HTTP GET requests.

The creation of names that include linguistically based constructs is strongly discouraged. Such names do not age or travel well, and names that look more or less like numbers avoid common problems that defeat persistence and international acceptance. The use of digits is highly recommended, mixed in with non-vowel alphabetic characters if compact names are desired.

Hyphens are always ignored in ARKs. Hyphens may be added to an ARK for readability, or during the formatting and wrapping of text lines, but they must be treated as if they were not present. Like the NMAH, hyphens are semantically inert in comparing ARKs for equivalence. Thus, for example, the following ARKs are equivalent:

```
ark:/12025/65-4-xz-321
ark:sneezy.dopey.com/12025/654--xz32-1
ark:/12025/654xz321
```

3.4. Naming Considerations

Names must be chosen with great care. Poorly chosen and managed names will devastate any persistence strategy, and they do not discriminate based on naming scheme. Whether a mistakenly re-assigned identifier is a URN, DOI, PURL, URL, or ARK, the damage -- failed access -- is not mitigated more in one scheme than in another. Conversely, properly managed names will go much further towards safeguarding persistence than any choice of naming scheme or its

underlying protocols.

Similarly, hostnames appearing in any identifier meant to be persistent must be chosen with extreme care. While this certainly

affects names that are based on URLs and PURLs, it should be of concern in selecting names even for explicitly disposable entities such as the NMAHs that serve as temporary "booster rockets" to help make ARKs actionable. There is no excuse for a provider that manages its internal names impeccably not to apply the same skill to choosing an exceptionally durable internet name (e.g., a generic name in the ".org" domain), especially if it would form the prefix for all the provider's URL-based external names.

Dubious persistence speculation should be avoided. For example, there are really no obvious reasons why the organizations registering DNS names, URN NIDs, and DOI publisher IDs should have among them one that is intrinsically more fallible than the next. Moreover, it is a misconception that the demise of DNS and of HTTP need adversely affect the persistence of URLs. Certainly URLs from the present day would not then be actionable by our present-day mechanisms, but there is no more stable a namespace than one that is dead and frozen, and resolution systems for future non-actionable URLs are no harder to imagine than for currently non-actionable URNs and DOIs. The important point, however, is that just because hostnames have been carelessly used in their brief history does not mean that they are unsuitable in NMAHs (and URLs) intended for use in situations that demand the highest levels of persistence. A well-considered naming strategy is everything.

4. Assigners of ARKs

A Name Assigning Authority (NAA) is an organization that creates (or delegates creation of) long-term associations between identifiers and information objects. Examples of NAAs include national libraries, national archives, and publishers. An NAA may arrange with an external organization for identifier assignment. The US Library of Congress, for example, allows OCLC (the Online Computer Library Center, a major world cataloger of books) to create associations between Library of Congress call numbers (LCCNs) and the books that OCLC processes.

An NAA does not so much create an identifier as create an association. The NAA first draws an identifier from its namespace, which is the set of all identifiers under its control. It then records the assignment of the identifier to an information object having sundry witnessed characteristics, such as a particular author and modification date. A namespace is usually reserved for an NAA by agreement with recognized community organizations (such as IANA and ISO) that all names beginning with a particular string be under its control. In the ARK an NAA is represented by the Name Assigning Authority Number (NAAN).

The ARK namespace reserved for an NAA is the set of names bearing its particular NAAN. For example, all strings beginning with "ark:/12025/" are under control of the NAA registered under 12025, which might be the US National Library of Medicine. Each NAA is free

to assign names from its namespace (or delegate assignment) according to its own policies. These policies must be documented in a manner similar to the declarations required for URN NID registration.

[The details of ARK NAA registration have not been worked out. The next section has what might pass for informal registration.]

5. Finding a Name Mapping Authority

In order to derive an actionable identifier from an ARK, a host-port (hostname or hostname-port combination) for a working Name Mapping Authority (NMA) must be found. An NMA is a service that is able to respond to the three basic ARK service requests. NMAs make known which NAAs' identifiers they are willing to service.

Upon encountering an ARK, a user (or client software) looks inside it for the optional NMAH part. If it contains an NMAH, and if the user trusts it, and if the service works, the NMAH discovery step may be skipped. Otherwise, the client looks inside the ARK again for the NAAN (Name Assigning Authority Number). Then, using a global database, it uses the NAAN to look up all current NMAHs that service ARKs issued by the identified NAA.

In the interests of long-term persistence, ARK mechanisms are defined in high-level, protocol-independent terms so that mechanisms may evolve and be replaced over time without compromising fundamental service objectives. Such is the case then for mapping authority discovery and the two specific NMAH lookup methods outlined below. Either or both methods may eventually be supplanted by better methods since, by design, the ARK scheme does not depend on a particular method, but only on having some method to locate an active NMAH.

At the time of issuance, at least one NMAH for an ARK should be prepared to service it. That NMA may or may not be administered by the Name Assigning Authority (NAA) that created it. Consider the following hypothetical example of providing long-term access to a cancer research journal. The publisher needs to recover costs and the National Library of Medicine needs to preserve the scholarly record. By agreement the publisher would act as the NAA and the national library would archive the journal issue when it appears, but without providing direct access for the first six months. During the first six months of peak commercial viability, the publisher would retain exclusive delivery rights and would charge access fees. Again, by agreement, both the library and the publisher would act as NMAs, but during that initial period the library would redirect requests for issues less than six months old to the publisher. At the end of the waiting period, the library would then begin servicing requests for issues older than six months by tapping directly into

its own collection. Meanwhile, the publisher might redirect incoming requests for old back issues to the library. Long-term access is thereby preserved, and so is the commercial incentive to publish content.

There is never a requirement that an NAA also run an NMA service, although it seems not an unlikely scenario. Over time NAAs and NMAs will come and go. One NMA will succeed another, and there may be many NMAs serving the same ARKs simultaneously. These may be mirrored NMAs, competing NMAs, or asymmetric but coordinated NMAs, as in the library-publisher example above.

5.1. Looking Up NMAHs in a Globally Accessible File

This section describes a way to look up NMAHs using a simple text file. For efficient access the file may be stored in a local filesystem, but it should be reloaded periodically to incorporate updates. It is not expected that the size of the file or frequency of update should impose an undue maintenance burden any time soon. It is modeled on the /etc/hosts file mechanism that supported internet host address lookup for several years before the advent of the Domain Name System (DNS). A copy of the current file (at the time of writing) appears in an appendix and is available on the web. It looks like this, with comment lines (lines that begin with `#') explaining the format.

```
#
# Name Assigning Authority / Name Mapping Authority Lookup Table
#     Last change:  22 February 2001
#     Reload from:  http://xxx.xxx.xxx/etc/naa2nma.txt
#
# Each NAA appears at the beginning of a line with the NAA Number
#     first, then a colon and the name of the NAA organization
#
# All the NMA Host-ports that service an NAA are each listed,
#     one per line, indented, after the corresponding NAA line.
#
121:    US Library of Congress
        foobar.zaf.org
        sneezy.dopey.com
#
122:    US National Library of Medicine
        lhc.nlm.nih.gov:8080
        foobar.zaf.org
        sneezy.dopey.com
#
123:    US National Agriculture Library
        foobar.zaf.gov:80
#
# The enclosed Perl script takes an NAA as argument and prints
# the NMAs in this file listed under the first matching NAA.
#
# my $naa = shift;
```

```
# while (<>) {
#         next if (! /^$naa:/);
#         while (<> && /^) { print "$10 if (/^; }
# }
# end of file
```

5.2. Looking up NMAHs Distributed via DNS NAPTR Records

This section sketches a way to look up NMAHs using a method very similar to the method, described in [RFC2168], for locating URN resolvers. It relies on querying the DNS system already installed in the background infrastructure of most networked computers. A query is submitted asking for a list of resolvers that match a given NAAN. The query is distributed via normal DNS channels to the particular DNS servers that can best provide the answer, unless the answer is already in a local DNS cache as a side-effect of a recent query. Responses come back inside Name Authority Pointer (NAPTR) records. The result is zero or more candidate NMAHs.

[Details to follow in a revised draft.]

6. Generic ARK Service Definition

Again, ARK services must be couched in protocol-independent terms if persistence is to outlive today's infrastructural assumptions. The high-level ARK service definitions listed below are followed in the next section by a concrete method (one of many possible methods) for delivering these services with today's technology. An ARK request's output is delivered information, such as the object itself, a policy declaration (e.g., a promise of support), a descriptive metadata record, or an error message.

6.1. General ARK Access Service (access, location)

Returns (a copy of) the object or a redirect to the same. A sensible object surrogate may be substituted; for example, a table of contents for a large complex object, a home page for a web site hierarchy, or a rights clearance challenge for protected data. May also return a discriminated list of alternate object locators. If access is denied, returns an explanation of the object's current (perhaps permanent) inaccessibility.

6.2. General Policy Service (permanence, naming, addressing)

Returns declarations of policy and support commitments for given ARKs. Declarations are returned in either a structured metadata format or a human readable text format (one format may serve both purposes). Policy areas may be returned separately, but covered areas should including object permanence, object naming, object fragment addressing, and operational service support.

The permanence declaration for an object is a rating defined with respect to the identified permanence guarantor, an includes the following aspects.

(a) "object availability" -- whether and how access to the object is supported (e.g., online 24x7, or offline only),

(b) "identifier validity" -- under what conditions the ARK will be or has been re-assigned,

(c) "content invariance" -- under what conditions the content of the object is subject to change, and

(d) "link stability" -- whether and how the hypertext links within and among parts of the object are subject to change.

Naming policy for an object includes an historical description of the NAA's (and its successor NAA's) policies regarding differentiation of objects. It includes the following aspects.

(a) "similarity" -- the set of criteria, defined by the NAA, at which point two similar objects become dissimilar enough to warrant separate ARKs, and

(b) "granularity" -- the limit, defined by the NAA, to the level of object subdivision below which sub-objects do not warrant separately assigned ARKs.

Addressing policy for an object includes a description of how, during access, object components (e.g., paragraphs, sections) or views (e.g., image conversions) may or may not be "addressed", in other words, how the NMA permits arguments (or parameters) to modify the object delivered as the result of an ARK request. These sorts of operations, if allowed, would support things like byte-ranged fragment delivery and open-ended format conversion too numerous to list, let alone to identify with many separately assigned ARKs.

Support policy includes a description of general operational aspects of the NMA service, such as after hours staffing and trouble reporting procedures.

6.3. General Description Service

Returns a description of the object. Descriptions are returned in either a structured metadata format or a human readable text format (one format may serve both purposes). A description must at a minimum answer the who, what, when, and where questions concerning an expression of the object. A description must always be accompanied by the identity of the description's source and the its modification date. May also return discriminated lists of ARKs that are related to the given ARK.

7. Overview of the HTTP Key Mapping Protocol (HKMP)

The HTTP Key Mapping Protocol (HKMP) is a way of taking a key (an

identifier) and asking such questions as what information does this identify and how permanent is it? HKMP is in fact one specific method for delivering ARK services. It runs over HTTP in order to exploits HTTP's simplicity and the pre-eminence of the web browser

user interfaces that rely on it. The method is designed so that an asker (a person or client program) can enter ARK commands directly into the location field of current browsers.

The asker starts with an identifier, such as an ARK (or a URL). The identifier reveals to the asker (or should allow the asker to infer) the internet host name and port number of a server system that responds to questions. This is just the NMAH that is obtained by inspection, and possibly lookup based on the ARK's NAAN. The asker then sets up an HTTP session with the server system, sends a question via an HKMP request (contained within an HTTP request), receives an answer via an HKMP response (contained within an HTTP response), and closes the session. That concludes the connected portion of the protocol.

An HKMP request is a string of characters that is appended to the identifier string after first adding a `?' (question mark). The resulting string is sent as an argument to HTTP's GET command. Request strings too long for GET may be sent using HTTP's POST command.

An HKMP response is contained in the headers and message body of the HTTP response. The headers convey a few parameters relevant to the HKMP process and the HTTP message body consists of a set of one or more records. Records themselves tend to look very similar in format to HTTP response headers (also very similar to email headers), with records separated by empty lines. Precise record formatting rules and semantics are defined as for Electronic Resource Citations [ERC]. [Although ERCs are still work in progress, a sense of how they work may be obtained from the examples below.]

Here's an example using an HTML document associated with a key given by the hypothetical URL,

http://foo.bar.org:8080/12025/543cb

The asker prepares to request that a copy of the information resource associated with the key be returned by appending to it the HKMP request,

?get(it)

The entire client-server session looks as follows, and can easily be conducted by keyboard using an ordinary [TELNET] program.

```
C: [opens session]
C: GET http://foo.bar.org:8080/12025/543cb?get(it) HTTP/1.1
C:
S: HTTP/1.1 200 OK
```

```
S: Content-Type: text/html
S: HKMP-Status: 200 OK
S:
S: <HTML>
```

```
S: ... (text of document) ...
S: </HTML>
S: [closes session]
```

The first and last lines correspond to the client's steps to start a [TCP] session with the server (e.g., starting up a TELNET program directed at the server's host and port) and the server's steps to end that session, respectively. Although they are needed for each session, to keep things simple future examples will omit them.

The first two server response lines above are typical of HTTP. The next line is peculiar to HKMP, and indicates a normal return status. The remainder of the response is the stream of HTML that comprises the returned document. The particular request serviced in this example is so common that it is taken as the default request in the event that the asker appends nothing at all to the identifier string. Thus naked identifiers carried over HKMP end up requesting straightforward object access.

In asking for a citation for the information associated with a key, the following session might be conducted. Here we assume that the client initiates the session and the server terminates it. The appended HKMP request is simply "?get(cite)" and the returned data is a single ERC record. Because this request is so common, the asker can simply append the HKMP request "?" to get equivalent behavior.

```
C: GET http://foo.bar.org:8080/12025/543cb?get(cite) HTTP/1.1
C:
S: HTTP/1.1 200 OK
S: Content-Type: text/plain
S: HKMP-Status: 200 OK
S:
S: erc:
S: who: Dr. Seuss
S: what: Green Eggs and Ham
S: when: 1962
S: where: http://nma.seussfans.org/geah.txt
S: erc-about:
S: what: poultry products | pork products
S:          | fear of the unknown
S: erc-from:
S: who: NLM
S: what: ui12345678
S: when: 2000 11 09
S: where: http://nma.nih.gov/ui12345678?%{
S:         db = books
S:         & param1 = 259
S:         & xyz = tt25_90
```

```
S:          & type = basic_cite + topics
S: %}
```

Several ERC features are worth noting. If there were more than one record, each would be separated from the next by an empty line. Each

ERC itself may be organized in different segments according to the different stories each segment tells. In the above example, the first story concerns an expression of an object that happens to be a children's book. The second story (containing but one element) concerns what the object is about. The third and final story concerns the provenance (or origin) of the ERC record itself.

The ERC also has an appealing set of uniform lexical features. Long elements may be continued on subsequent indented lines and there is a standard way to indicate subelement boundaries. Especially long, dense strings (e.g., some URLs) can be leavened with whitespace (newlines, spaces, tabs) through an encoding trick that makes them easier for human being to read and edit; of course, decoding restores the string to its original form. The date format accommodates lists and ranges, and any element may use a special marker syntax to indicate that its data value has been inverted to create a sequence of bytes that forms a better sorting point; the marker syntax can be used to indicate how to recover the natural sequence of bytes.

As a final example, here is a session in which an asker requests a permanence declaration associated with a key. The returned data is in the form of an ERC because the HKMP request is "?get(permanence)". If the request were instead "?show(permanence)", the server would be instructed to return a declaration formatted for human display; one server might still return an ERC, but another might return an HTTP Content-Type of text/html followed by an HTML document that describes the policy declaration in natural language.

```
C: GET http://a.b.org/92284/6xz21?get(permanence) HTTP/1.1
C:
S: HTTP/1.1    200    OK
S: Content-Type: text/plain
S: HKMP-Status: 200 OK
S:
S: erc-permanence:
S: who: LC
S: what: (:psc) Permanent, Stable Content
S: when: 1997 12 04
S: identifier_validity: Permanent
S: content_invariance: Subject to Correction
S: resource_availability: Always Available
S: erc-from:
S: who: NLM
S: what: ui9876543
S: when: 1999 03 24
S: where: http://nma.nih.gov/ui12345678?pstmt
```

8. Advice for Web Clients

This section offers some advice to web clients. It is hard to write about because it tries to anticipate a series of events that may (or may not) lead to native web browser support for ARKs.

ARKs are envisaged to appear wherever durable object references are planned. Library cataloging records, literature citations, and bibliographies are important examples. In many of these places URLs (Uniform Resource Locators) currently stand in, and URNs, DOIs, and PURLs have been proposed as alternatives.

The strings representing ARKs are thus also envisaged to appear in some of the places where URLs appear: inside hypertext links where they are not normally shown to users and as manifest text where the ARK itself may be of interest (e.g., printed in a document footer, or in search results). In many cases both the effect of a hypertext link and the manifest link are of interest, as is typically found in the results from the internet search engines. A normal URL-based HTML link looks like

```
<a href = "http://a.b.org/index.htm"> Click Here <a>
```

The same link with an ARK instead of a URL looks like

```
<a href = "ark:/14697/b12345x"> Click Here <a>
```

Web browsers would in general require a small modification to convert this kind of ARK into a specified-host ARK, as in

```
<a href = "ark:a.b.org/14697/b12345x"> Click Here <a>
```

whence a trivial browser modification is required to treat this as

```
<a href = "http://a.b.org/14697/b12345x"> Click Here <a>
```

No browser modification is required if the prefix is omitted, as in

```
<a href = "a.b.org/b12345x"> Click Here <a>
```

because the prefix "http://" is generally assumed.

An NAA will typically make known the associations it creates by publishing them in catalogs, actively advertizing them, or passively leaving them on web sites for visiting indexing spiders.

A valuable technique for provision of persistent objects is to try to have the identifier appear on, with, or near its retrieved object. An object could then easily be traced back to its metadata, to alternate versions, to updates, etc. This has seen reasonably widespread success, for example, in software distributions.

9. Security Considerations

The ARK naming scheme poses no direct risk to computers and networks. Implementors of ARK services need to be aware of security issues when querying networks and filesystems for Name Mapping Authority services, and the concomitant risks from spoofing and obtaining

incorrect information. These risks are no greater for ARK mapping authority discovery than for other processes of service discovery. For example, recipients of ARKs with a specified hostport (NMAH) should treat it like a URL and be aware that the identified ARK service may no longer be operational.

Similarly, ARK clients and servers subject themselves to all the risks that accompany normal operation of the protocols (e.g., HTTP, Z39.50) underlying mapping services. As specializations of such protocols, a concrete ARK service may actually limit exposure to their usual risks. Indeed, ARK services may enhance security by helping users identify long-term reliable references to information objects. On the other hand, due to extreme age of some ARK identifiers, the chances may be higher of coming across systems far enough out of the mainstream that spoofing will be harder to detect.

10. Acknowledgements

The ARK scheme would not have come to maturity without many long and in-depth conversations with Rick Rodgers. The generous support of the US National Library of Medicine (NLM) is gratefully acknowledged, as are the very stimulating discussions I had with members of NLM's Permanence Working Group.

11. Author's Address

John A. Kunze
Center for Knowledge Management
University of California, San Francisco
530 Parnassus Ave, Box 0840
San Francisco, CA 94143-0840, USA

Fax: +1 415-476-4653
EMail: jak@ckm.ucsf.edu

12. References

[RFC2168] Daniel, R. and M. Mealling, "Resolution of Uniform Resource Identifiers using the Domain Name System", RFC 2168, June 1997.

[RFC2169] Daniel, R., "A Trivial Convention for using HTTP in URN Resolution", RFC 2169, June 1997.

[RFC2141] Moats, R., "URN Syntax", RFC 2141, May 1997.

[RFC1737] Sollins, K. and L. Masinter, "Functional Requirements for Uniform Resource Names", RFC 1737, December 1994.

[RFC2396] T. Berners-Lee, R. Fielding, L. Masinter, "Uniform

Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[RFC2616] R. Fielding, J. Gettys, J. Mogul, et al, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[ERC] J. Kunze, "Electronic Resource Citations", work in progress

13. Appendix: Current /etc/arkna File

```
#
# Name Assigning Authority / Name Mapping Authority Lookup Table
#     Last change:  22 February 2001
#     Reload from:  http://xxx.xxx.xxx/etc/naa2nma.txt
#
# Each NAA appears at the beginning of a line with the NAA Number
#     first, then a colon and the name of the NAA organization
#
# All the NMA Host-ports that service an NAA are each listed,
#     one per line, indented, after the corresponding NAA line.
#
121:    US Library of Congress
    foobar.zaf.org
    sneezy.dopey.com
#
122:    US National Library of Medicine
    lhc.nlm.nih.gov:8080
    foobar.zaf.org
    sneezy.dopey.com
#
123:    US National Agriculture Library
    foobar.zaf.gov:80
#
# The enclosed Perl script takes an NAA as argument and prints
# the NMAs in this file listed under the first matching NAA.
#
# my $naa = shift;
# while (<>) {
#         next if (! /^$naa:/);
#         while (<> && /^) { print "$10 if (/^; }
# }
# end of file
```

14. Copyright Notice

Table of Contents