

# Actionable Resource Tags for Virtual Organizations

Joe Futrelle<sup>1</sup>

National Center for Supercomputing Applications  
1205 W. Clark St., Urbana IL 61801, US  
futrelle@uiuc.edu

**Abstract.** Locating artifacts produced in complex, distributed processes (e.g., Virtual Organizations) is made difficult by the requirement that artifacts be uniquely identified. Many global identification schemes (e.g., handles) employ a central locus of control (e.g., a naming service) to guarantee uniqueness while permitting distributed generation of unique identifiers. However, as [1] has noted, this kind of identifier is no more persistent than the services that manage it. ARK[1] and the tag URI scheme[2] both provide means of relaxing the requirement for central loci of control; this report attempts to synthesize the best features of both schemes into a new proposal.

## 1 Theoretical background

As the number of digital artifacts produced and consumed daily continues to grow without apparent bound, robust identification schemes are required to guarantee some level of service as the artifacts age. A scheme should ideally ensure that unique identifiers can be produced with a minimum of coordination, and that they remain useful (i.e., they can be used to access the resources they identify) for as long as possible. A number of schemes exist to deal with one or both of these issues, but they typically require more central coordination than is feasible in VO's, or assume that today's IT environment (e.g., the web) will outlive the artifacts it is managing.

A number of recent proposed identifier schemes offer some useful strategies for improving the durability of identifiers in loosely-coupled environments. This report looks at the most useful strategies from ARK[1] and the tag URI scheme[2] and attempts to synthesize them into a new proposal.

### 1.1 Persistence, assignment and mapping

As [1] points out, the idea that an identifier for some resource is “persistent” is a gloss that generally means one or both of the following:

1. the identifier names one and only one resource, and/or
2. the resource will always be accessible given its identifier.

These are very different guarantees. The first constrains the processes that generate identifiers, which must ensure that every identifier generated is unique. The second constrains the processes that locate resources by identifier, which must ensure that mappings between identifiers and resources are immutable. In [1], these two kinds of processes are called “name assignment” and “name mapping,” and that terminology is adopted for the rest of this report.

## 1.2 Name assignment

From the perspective of a name-assignment process, the question “can I assign a name in namespace X?” amounts to “who controls namespace X?” In closed systems (e.g., autoincrement columns in relational databases) control over namespaces is often subsumed by local scoping mechanisms. Outside of closed systems, most schemes use hierarchical namespaces backed by some service and/or manual process (e.g., DNS). The problem with that strategy is that the services and manual processes required to partition the global namespace into sub-namespaces in one of these schemes may not outlive the names so generated. For instance `http://www.foo.com/index.html` might refer to my web page today, but I may sell `foo.com` to someone else tomorrow, who could then replace my page with different content.

Claiming that a name in some namespace is unique specifically implies that whoever assigned the name gained control of the namespace at some point in time and retained it at least until the name was assigned. But when? The tag URI scheme makes the temporal scope of a uniqueness claim explicit in its concept of a `taggingEntity`:

The tagging entity is designated by an “authority name” – a fully qualified domain name or an email address containing a fully qualified domain name – followed by a date. The date is chosen to make the tagging entity globally unique, exploiting the fact that domain names and email addresses are assigned to at most one entity at a time. That entity then ensures that it mints unique identifiers[2].

A `taggingEntity` is roughly equivalent to a name assignment authority in ARK, with the advantage that no central process is required to ensure that tagging entities are unique (so long as the processes claiming to be tagging entities are honest). Of course, DNS is a hierarchical process, but a person’s ownership of their email address or an organization’s control of a DNS name at any given point in time is typically a fact at hand.

## 1.3 Name mapping

Locators (e.g., URL’s) are identifiers that identify not only a resource but also how to take action to access the resource. In general, locators cannot survive the migration of a resource from one access modality to another. Adding a level of indirection, as with PURL, mitigates but does not solve this problem, since

an indirect locator cannot be resolved once the indirection service it directly identifies is no longer available (e.g., when `purl.org` is no more). Put differently, locators are poor candidates for persistent resource identifiers because they make no distinction between name assignment and name mapping.

Name mapping is simply the association of identifiers with resources, and conceptually that is a class of service that provides access to a resource given its identifier. In ARK, instances of this class of service are called “name mapping authorities.”[1]. ARK not only gives different syntactic status to name mapping authorities and name assignment authorities, but explicitly excludes name mapping authorities from the identity condition for ARK’s:

The [Name Mapping Authority Host/port] is thus a kind of identity-inert, disposable booster rocket that launches the ARK into cyber-space while allowing for limited branding. When the Web no longer exists, the core identity of the ARK is easily recovered by isolating the part of the ARK that begins with `ark:/` [1].

The exclusion of the NMAH from the identity condition for ARK means that the resource named by an ARK can migrate to different NMAH’s at different times while retaining its identity, and also that the identity of an ARK is not predicated on the continued serviceability of any given NMAH. This is called an “actionable identifier”[1] to distinguish it from locators in general, which are actionable but identify both a resource and an access point. Importantly, actionable identifiers can be assigned without cooperation between the NAA and NMA’s, since the only namespace that participates in the identity of an actionable identifier is its NAA’s.

ARK further specifies three kinds of actionability for every ARK that begins with an NMAH, with a syntax for each one:

1. Visiting the ARK in a browser or other web agent returns the resource the ARK identifies
2. Appending a ‘?’ to the ARK and visiting the resulting URL in a browser or other web agent returns metadata about the resource
3. Appending a ‘??’ to the ARK and visiting the resulting URL in a browser or other web agent returns a “commitment statement” characterizing the NMAH’s retention policy for the resource and/or name mapping

ARK could alternatively have solved problem of distinguishing retention policy from descriptive metadata by having only one metadata access point and making the metadata description rich enough to distinguish between the two kinds of information.

#### 1.4 Names

Within a namespace, what names should be allowed? Most identifier schemes specify a production and then allow anything in that production. This usually allows information about a resource to be encoded in its name, a practice with

known preservation risks. The California Digital Library’s use of ARK is based on some interesting observations about maintaining name opacity over time:

The [California Digital Library] also takes steps to avoid accidental semantics. It restricts the ARK character set to digits and non-vowels, doesn’t create identifiers with more than two non-digits in a row, generates a check character along with each ARK (which guarantees the ARK against the most common single character and transposition errors), and even uses a kind of pseudo-random sequencing to avoid “series semantics” that could reveal the order of identifier assignments [1].

By contrast, the `tag` URI specification recommends that the `specific identifier` be “human-friendly,” which presumably recommends against the kind of opacity described above.

## 2 Proposed scheme

The following proposal describes an identifier scheme for Actionable Resource Tags, or ART’s. The scheme is designed to combine the opacity and actionability of ARK’s with the lightweight name assignment of tag URI’s.

### 2.1 Requirements

The ART scheme is built around the following requirements:

1. Each ART is globally unique
2. Each ART is opaque, e.g., it contains no information about the resource it identifies
3. Each ART is scoped to a `taggingEntity` identifying a name assignment authority
4. Each ART may begin with the URL of a Name Mapping Authority Host/port

**Global uniqueness.** Global uniqueness of identifiers is impossible to enforce without more knowledge than is ever available, so ART’s provide strong mechanisms only for enforcing the global uniqueness of namespaces. Namespaces are equivalent in the ART scheme to name assignment authorities, which are identified using the `taggingEntity` production from the tag URI scheme[2]. As with ARK, the identity of an ART is predicated only on its NAAI and Name; the NMAH, if present, must be ignored when performing comparisons.

**Opacity.** Identifiers should not encode information about the resources they identify. They also shouldn’t appear to do so. Since neither of these requirements can be enforced at the syntactic level, a highly-constrained production along with recommendations on how to avoid the appearance of informativeness will have to suffice.

**Actionability.** As in ARK[1], any ART may begin with the URL of an NMAH. The resulting locator is then expected to deliver a browser or other internet agent to the resource identified by the ART. Appending a ‘?’ to an ART will deliver the agent to metadata about the resource; ART does not constrain the representation of this metadata in any way, but recommends that web agents use content negotiation to request the metadata in a representation that they understand.

## 2.2 Syntax

This section describes the ART syntax, which is closely patterned after ARK. An ART consists of four parts:

1. An optional NMAH URL, e.g. “`http://www.foo.com/`”
2. The ART scheme identifier followed by a slash, i.e. “`art:/`”
3. A Name Assignment Authority Identifier in the form of a `taggingEntity`, followed by a slash, e.g., “`futrelle@uiuc.edu,1997/`”
4. A name, e.g., “`v82n0p1s`”

**Names.** Names in ART can contain only lowercase roman letters and arabic numerals. It is recommended that names be chosen so that they do not contain words or commonly-used acronyms. Sequences of digits should be chosen so that they don’t appear to represent dates.

**Compatibility with ARK.** An ARK can be transformed into an ART by changing the scheme from `ark` to `art` and appending “`@naan.cdlib.org,2003`” to the NAAN. It is not possible to transform an ART into an ARK unless the ART’s NAAI is associated with a NAAN in CDL’s global NAAN registry.

**Compatibility with tag URI’s.** An ART can be transformed into a tag URI by stripping off the NMAH and scheme identifier and replacing them with `tag::`; and replacing the slash separating the NAAI from the Name with a colon. A tag URI cannot be transformed into an ART unless its `specific` part conforms to the ART Name production or can be mapped into that production in a way that preserves the uniqueness of the other names minted by that `taggingEntity`.

**Compatibility with URI’s and URL’s.** All ART’s are URI’s. ART’s that begin with an NMAH are also URL’s.

## References

1. Kunze, J. A. “Towards Electronic Persistence Using ARK Identifiers.”
2. “Tag URI Scheme,” IETF RFC 4151. <http://www.faqs.org/rfcs/rfc4151.html>