ARK Identifiers FAQ, version 0.91

THIS IS AN OLD VERSION. HERE IS THE LATEST ARKs FAQ.

Basics

How can I give feedback on this document?

By inserting comments into this comment-friendly version.

What are ARKs?

ARKs (Archival Resource Keys) are high-functioning *identifiers* that lead you to things and to descriptions of those things. For example, this ARK,

`https://n2t.net/ark:/67531/metadc107835/`

gets you to a dissertation, and adding a '?' on the end of the ARK should get you to its description:

`https://n2t.net/ark:/67531/metadc107835/?`

What's an identifier?

On the internet, an *identifier* is a URL, or part of a URL. For example, this core ARK identifier,

`ark:/12148/btv1b8449691v/f29`

appears inside two different URLs (Uniform Resource Locators, also known as web links or web addresses):

`https://gallica.bnf.fr/`**`ark:/12148/btv1b8449691v/f29`**

`https://n2t.net/`**`ark:/12148/btv1b8449691v/f29`**

ARKs are especially good at being *persistent identifiers*.

What's a persistent identifier?

The average lifetime of a URL was once said to be 44 days. At the end of its life, a URL link *breaks*, meaning it gives you the dreaded "404 Not Found" error that most of us have seen. Irritating as that may be, it's politically awkward when looking for publicly funded research, and it's a cultural disaster for libraries, archives, museums, and other memory organizations.

A *persistent identifier* is a link that in principle keeps working far into the future, even as things move between websites. Normally when things move, everyone who ever recorded the old links would need to be told what the new links are, which is next to impossible. That's where identifier *resolvers* come in.

What's a resolver?

A *resolver* is a website that specializes in forwarding incoming identifiers (the ones originally advertised to users) to whichever websites are currently best able to deal with them. Overall, forwarding is called *resolution*; one step in a resolution process is called *redirection*.

For a resolver to work, its hostname (the `n2t.net` or `gallica.bnf.fr` in the identifiers above) must be carefully chosen so it won't ever need to be changed. Memory organizations, some of them centuries old, tend to have hostnames well-suited to be resolvers. Some well-known, younger resolvers are n2t.net (the ARK resolver), identifiers.org, doi.org, handle.net, and purl.org.

What are ARKs used for?

For anything and everything. Current uses of ARKs include

- digital content, such as genealogical records (FamilySearch)
- publisher content (Portico)
- digitized manuscripts (Gallica)
- texts (Internet Archive)
- museum holdings (Smithsonian)
- vocabulary terms (yamz.net, perio.do)
- historical figures (snaccooperative.org)
- datasets, journals, living beings, and more.

Who is using ARKs?

That's a little hard to say because ARKs are very decentralized, but more than 500 registered organizations have, between them, created an estimated 3.2 billion ARKs. Here's who is listed the public registry.

Getting started

What do I need to create ARKs?

First you need a *NAAN* (Name Assigning Authority Number), which is a number reserved exclusively for your organization. It must appear in every ARK your organization assigns, just after the "`ark:/`" label. The NAAN in all of these ARKs,

`ark:/12148/btv1b8449691v/f29`

`https://gallica.bnf.fr/`**`ark:/12148/btv1b8449691v/f29`**

`https://n2t.net/`**`ark:/12148/btv1b8449691v/f29`**

is **`12148`**, and it uniquely identifies the French National Library. Each NAAN is associated with the URL of a resolver for its ARKs, for example, to resolve `12148` ARKs, append them to `https://gallica.bnf.fr/` as shown in above. The N2T.net resolver is unusual in that it routes any ARK to the resolver registered under its NAAN.

There is no charge to obtain or use a NAAN, and you can request one by filling out an online form. Over 500 organizations have NAANs – libraries, archives, museums, university departments, government agencies, scholarly and educational publishers, etc. – all listed in the public NAAN registry.

What kinds of things can I assign ARKs to?

To anything digital, physical, or abstract. That can include things that *don't yet exist* but to which you need to refer from objects that you're in the process of creating or planning, such as a link from a draft article to a dataset under preparation, or a link from an archived digital letter to a planned finding aid.

One caution is that you should generally assign ARKs to things that you own, control, or manage. Assigning ARKs to things you don't control is discouraged because such identifiers tend to be fragile.

How do I start creating the character strings that become ARKs?

You are free to create ARK strings as you wish, provided you use only digits, letters upper- and lower-case letters (ASCII, no diacritics), and the following characters:

```
= ~ * + @ _ $ . /
```

The last two characters are reserved in the event you wish to disclose ARK relationships. A unique feature of ARKs is that hyphens ('-') may appear but are *identity inert*, meaning that strings that differ only by hyphens are considered identical; for example, these strings

```
ark:/12345/141e86dc-d396-4e59-bbc2-4c3bf5326152
```

```
ark:/12345/141e86dcd3964e59bbc24c3bf5326152
```

identify the same thing. The reason for this feature is that text formatting processes out in the world routinely introduce extra hyphens into identifiers, breaking links to any server that treats hyphens as significant.

Regarding assignment strategy, it is common to leverage legacy identifiers. For example, a museum moth specimen number `cd456_f987` might be advertised under the `ark:/12345/cd456_f987`. Some legacy identifiers may need to be altered in view of ARK character restrictions.

The second common strategy is to make up entirely new strings for your ARKs. In this case it is important to consider whether to make them *opaque* or non-opaque (or a bit of both).

What are opaque identifiers?

Persistent identifier strings are typically *opaque*, deliberately revealing little about what they're assigned to, because non-opaque identifiers do not age or travel well. Organization names are notoriously transient, which is why NAANs are opaque numbers. As titles and dates are corrected, word meanings evolve (eg, innocent older acronyms may become offensive or infringing), strings meant to be persistent can become confusing or politically challenging. The generation and assignment of completely opaque strings comes with risk too, for example, numbers assigned sequentially reveal timing information and strings containing letters can unintentionally spell words.

Example strings with a range of opacity			
non-opaque	Netscape Permanent Archive	Gay_Divorcee_1934_April_1	Name-to-Thing Resolver
opaque-ish	x0001, x0002, ..., x9998	GD/1934/04/01	n2t.net
opaquer	141e86dc-d396-4e59-bbc2-4c3bf5326152	19340401	n2t
opaquest	141e86dcd3964e59bbc24c3bf5326152	h8k74926g	12148

ARKs are not required to be opaque, but it is recommended that the base object name be made opaque, since it tends to name the main focus of persistence. If any qualifier strings follow that name, it may be less important that they be opaque. To help decide what your approach to opacity, you may wish to consider compatibility with legacy identifiers and ease of string generation and transcription (eg, brevity, check digits). New strings can be created (minted) with date/time, UUID, and number generators, as well as Noid (Nice Opaque Identifiers) minters.

Opaque strings are "mute" and therefore challenging to manage, which is why ARKs were designed to be "talking" identifiers. This means that if there's ARK Identifiers FAQ, version 0.91#metadata, an ARK that comes in to your server with the '?' inflection should be able to talk about itself.

How do I make server content addressable with ARKs?

First, decide what the user experience of accessing your ARKs will be, for example, a spreadsheet file, a PDF, an image, a landing page filled with formatted metadata and a range of choices, etc. Whichever you choose, plan for your server to be able to respond with metadata if your ARK should arrive with a '?' inflection after it.

Otherwise, serving ARKs is like serving URLs. Normally incoming URL strings *address* (get mapped to) content that your web server returns. If your server is ARK-aware, incoming ARKs (expressed as URLs) must be mapped to the same content. A common approach is to map the ARK to the URL using a software table that you update whenever the URL changes. In this case your server is acting as a *local resolver*. If you don't want to implement this yourself, there are ARK software tools and services that can help.

Another approach is to run your web server without change, but instead of updating local tables, you would update ARK-to-URL mapping tables residing at a non-local resolver. Examples of this can be found among vendors and in any organization that updates tables via EZID.cdlib.org (which, due to a special relationship, updates resolver tables at n2t.net).

How do I advertise my ARKs?

An ARK meant for external use is generally advertised (released, published, disseminated) as an *actionable identifier*, in other words, inside a full URL starting with `https://` as in the examples above. A more compact visual display of an actionable ARK starting with `ark:/` can be achieved with an HTML hyperlink such as

```
<a href="https://n2t.net/ark:/99166/w66d60p2">ark:/99166/w66d60p2</a>
```

An important decision is whether your URL-based ARKs will use the hostname of your local resolver or the N2T.net resolver. If local control or branding is important enough, you would advertise ARKs based at your local resolver. If you're concerned about the stability of your local hostname, you would advertise your ARKs based at n2t.net (see examples of both).

Resolving your ARKs through N2T is always possible for users, regardless of how you advertise them.

Are there tools and services to help with ARKs?

Here's a partial list of software tools for persistent identification that includes

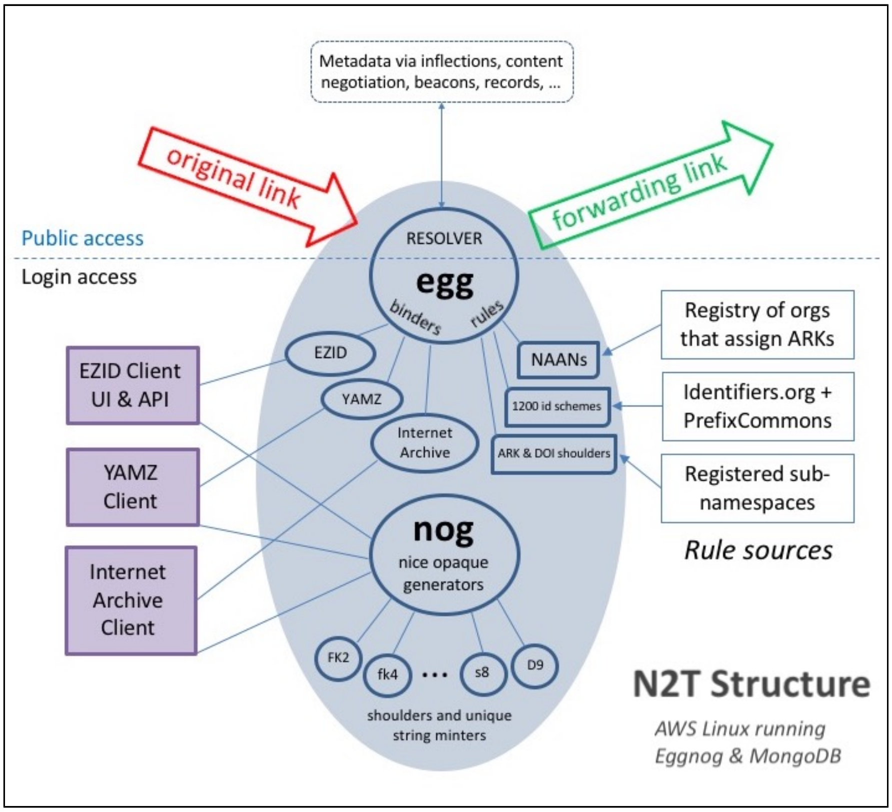
- Noid (Nice Opaque Identifiers), open source software for minting and resolving ARKs on your own
- ARK plugin for Omeka, which creates and manages ARKs for the Omeka open source web-publishing platform
- ARK module for Drupal, which allows your Drupal site to act as a Name Mapping Authority (NMA)

There are also some vendors, such as ezid.cdlib.org, and some more information on concepts and best practices.

Intermediate ARKs

What is N2T?

N2T.net is a global ARK resolver. N2T, which stands for Name-to-Thing, is actually a generalized resolver for mapping names into things, so it knows where to route over 600 other types of identifier – ARK, DOI, PMID, Taxon, PDB, ISSN, etc. If you're interested, the diagram and rest of this answer give a bit more detail.



N2T uses two kinds of stored data. First, it stores individual records for over 20 million object identifiers (eg, ARKs, DOIs) that it obtains from three sources: EZID.cdlib.org, Internet Archive, and YAMZ.net. When such records include a redirection URL (*target*) and descriptive ARK Identifiers FAQ, version 0.91#metadata, N2T can act on ARK Identifiers FAQ, version 0.91#inflections as well as perform suffix passthrough and "content negotiation".

Second, N2T stores over 3500 "rule" records for routing identifiers not found individually in N2T, but for which it has redirection information tied to the type of identifier being resolved. It obtains rule records from several sources, including the NAAN registry, a database of ARK and DOI *shoulders*, and a formal partnership on compact identifiers with identifiers.org.

What are the parts of an ARK?

```
 ARK ANATOMY                 Core
Immutable Identity
                          _____________
__|_________________
                         /                               \
       Resolver Service   Base Object Name    Qualifier
      _________|_________  ________|________  _______|_______
     /                   \/                 \/               \
     https://example.org/ark:12025/654xz321/s3/f8.05v.tiff
             \_________/ \__/\___/ \______/\____/\_______/
                 |        |    |      |      |       |
                 |      Label  |      |  Sub-parts  Variants
                 |             |      |
 Name Mapping Authority (NMA)  |   Assigned Name
                               |
               Name Assigning Authority Number (NAAN)
```

Can I assign ARKs to things inside something that already has an ARK?

Yes, ARKs can be assigned at any level of *granularity*, such as to a manuscript, to chapters inside it, to chapter sections, subsections, etc. An ARK can also be assigned to a thing that encloses other things. In ARKs the character '/' is reserved to help the recipient understand about containment, for example, the first ARK below contains the second ARK:

```
ark:/12148/btv1b8449691v

ark:/12148/btv1b8449691v/f29
```

That's the containment *qualifier*. There's only one other ARK qualifier, and it indicates *variant* forms of a thing by using the reserved character '.' in front of a suffix. For example, if these ARKs identify documents,

```
ark:/12148/btv1b8449691v/f29.pdf

ark:/12148/btv1b8449691v/f29.html
```

because they differ only by the suffix `.pdf` or `.html`, it can be inferred that they identify two different forms of the same document.

What is the purpose of the NAAN, and can I make changes to it?

NAANs subdivide the set of all possible ARKs (the ARK *namespace*). The subset of ARKs under a given NAAN can be further subdivided into *shoulders* (eg, 12345/x2, 98765/b4), which can make it easy to delegate autonomous ARK assignment to departments in a large organization. ARK resolution is loosely based on NAANs, but because organizations split, ARKs accommodate the namespace splitting problem by supporting management of a namespace by more than one organization. If you transition into or out of a vendor relationship, there is no impediment to taking your NAAN with you.

You can change a NAAN by filling out the same online form used for requesting a new NAAN. Example reasons for a change may include

- notifying N2T that your organization's contact person or resolver URL will change,
- updating your organization's name assignment policy (sample policy),
- requesting an additional NAAN for a significant new body of ARKs or new organizational division, and
- transitioning your NAAN to another organization that will carry on your work and take over your NAAN.

ARKs and other identifiers

Why would I use ARKs compared to, for example, DOIs?

- To keep costs down.
- To work with exactly the metadata you want.
- To be able to create identifiers without metadata.
- To be able to create an identifier even before your object exists.
- To have an identifier as soon as you create the first draft of your data.
- To hold that identifier private while the data and metadata evolve, and decide (maybe years) later, to publish or discard it.
- To retain that identifier upon publication, perhaps then assigning an additional identifier, such as a DOI.
- Because ARKs, built for generic application and not publication, fit naturally into identifying physical samples and field stations.
- Because they won't break as text formatting processes out in the world routinely introduce hyphens into identifiers.
- To be able to change vendor and/or infrastructure without having to coordinate database transfers with a central authority.
- To be able to create a larger number of identifiers of a given length, since mixed case letters permit denser strings.
- To be able to deal with the namespace splitting problem without losing control of your identifiers.
- To link identifiers to different kinds of nuanced persistence commitments.
- To be able to add queries (eg, ?lang=en) when resolving your identifiers.
- To use open infrastructure consistent with your organization's values.
- To link directly to the objects you value instead of to landing pages.
- To create one identifier that enables millions (suffix passthrough).
- To access convenient, full-function metadata via ARK Identifiers FAQ, version 0.91#inflections.

What does ARK have in common with DOI, Handle, PURL, and URN?

These are the major persistent identifier types (or schemes).

- All have been in existence since 2001 or before.
- All are found in places like the Data Citation Index , Wikipedia, and ORCID.org profiles.
- All give access to almost any kind of thing, whether digital, physical, abstract, person, group, etc.

They also have very similar structure, as seen in the examples below, consisting of four parts:

`https://n2t.net/ark:/99999/12345`
`https://doi.org/10.99999/12345`
`https://handle.net/10.99999/12345`
`https://purl.org/99999/12345`
`https://<various>/urn:99999:12345`
	1. the protocol (`https://`) plus a hostname,
	2. just for ARK and URN, there's also a label ("ark:" or "urn:"),
	3. the name assigning authority (`99999`, `10.99999`, or `99999`), which is the organization or group that created a particular identifier,
	4. and finally, the *name*, or local identifier, that it assigned (`12345`).

And they all have little effect on persistence. See 10 persistent myths about persistent identifiers.

Wait, are you saying ARK, DOI, Handle, PURL, and URN are useless?

No, that's too strong a statement. But let's keep these identifier schemes (types) in perspective.

- They all fail to stop the major causes of broken links: loss of funding, natural disaster, social upheaval, war, deliberate removal, human error, and provider neglect.
- They all require you, the end provider, to update forwarding tables as URLs change.
- They all identify content that is subject to change or removal on future visits.
- They all have identifiers that break regularly and in large numbers – many thousands and more.
- A non-trivial fraction of each scheme's identifiers did, and will, fail permanently, requiring forwarding to "tombstone" pages.
- They all rely on ordinary redirection built in to web servers since 1994 and provided for free by hundreds of URL shortening services.

Given how little the schemes do for you, when choosing one you'll likely want to consider factors such as cost, risk, and openness.

How do ARKs differ from identifiers like DOIs, Handles, PURLs, and URNs?

The short answer

ARKs are the only mainstream, non-siloed, non-paywalled identifiers that you can register to use in about 48 hours. DOIs, Handles, and PURLs require resolution and other services to come from their respective centralized systems (silos).

That's not to say that persistence is free. Making any identifier persistent burdens you, the provider, with the costs of content management, hosting, monitoring, and forwarding. You can do those things yourself or with help from a vendor. But with ARKs, just as with URLs, you will not be charged separately for your identifiers and you will not be locked in to a special-purpose resolution silo that also locks out other identifiers.

ARKs are unusual in being decentralized. While one *can* get resolution services from a global ARK resolver called n2t.net, over 90% of the ARKs in the world are published without reference to it. More than 500 registered organizations across the world have, between them, created an estimated 3.2 billion ARKs, and, as with URLs, no one has ever paid an identifier fee to create them. Of course *maintaining* them isn't free. It is never without cost to keep content access persistent in the long term, regardless of identifier type.

More differences between ARKs, DOIs, Handles, PURLs, and URNs

- Landing pages: Crossref and DataCite DOIs link to publisher landing pages constructed around but *not directly to* objects you care about, but ARKs can freely link *directly to* objects you care about, which is machine- and human-friendly since it does not require an extra human navigation step for common tasks such as
  - opening an article's PDF file for reading,
  - referencing an image file meant to be incorporated automatically inline into a document, and
  - citing a spreadsheet to be used for direct data analysis by software.
- DOIs, Handles, etc. do not support ARK-style ARK Identifiers FAQ, version 0.91#inflections that permit access to metadata regardless of whether an identifier points to an object or its landing page.
- Unlike DOIs and Handles, ARKs don't have metadata requirements. ARKs that haven't been released into the world are easy to delete.
- All things eventually pass, including hostnames and the web itself and the "`https://`" protocol. When that first part of the identifier ceases to have meaning, only ARKs and URNs will include the label (eg, "ark:") indicating the type of identifier that remains.
- For DOIs, Handles, and PURLs, you are required to use their respective resolvers. ARKs and URNs, permit you to use your own resolver.
- To create DOIs and Handles, you are required to pay a membership fee and, for DOIs, per-DOI charges. There are no fees for ARKs, PURLs, and URNs.
- To create Handles, you are required to install and maintain a local Handle server, which gives you another system to monitor, patch, and troubleshoot.
- Although you can use a local or vendor resolver for your ARKs and URNs, ARKs can be resolved via the global n2t.net resolver.
- The envisioned URN resolution infrastructure was never built, so URNs are currently resolved as URLs, and there is no designated global URN-as-URL resolver. In order to register to create URNs, you must apply for a URN namespace.
- ARKs have some unique features that support early object development: ARKs can be deleted, can be born with no metadata, and can exist with any metadata you care to store.

But if ARKs can be deleted, how can they be trusted?

It actually makes ARKs more trustworthy. The ability to delete is a vital part of healthy collection management that is denied to those non-ARK identifier types prohibiting deletion under the presumption that people, once they are asked to commit, won't make mistakes. People armed with identifier management software regularly turn simple human errors into large-scale mistakes, even at the threshold of commitment. By making it difficult to clean them up, we force systems to drag those messes forward in perpetuity.

While not immune to such mistakes, ARKs have the big advantage that they can be created and deleted in the shadows, independent of release, publication, or archival commitment.

Can an object have both an ARK and a DOI?

Yes. Sometimes having two identifiers is useful, although it can become confusing when it happens often. Many people start by assigning ARKs to each thing they create in order to have a stable reference right from the beginning, even before they know whether they want to publish it, let alone keep it.

Starting with an ARK, you benefit from being able to keep the original identifier from birth through to public release as the object and its metadata matures. For the subset of things that you end up wanting to publish in places that require DOIs, you can assign DOIs at publication time. If your ARK is stable and has basic metadata, you're already doing everything you need to have a proper DOI. This is a way in which ARKs support early object development.

In such a scenario, to reduce the burden of maintaining both identifiers you could register the DOI to redirect to the ARK. At the cost of maintaining just one identifier (the ARK), this would keep newly published links and links previously stored and bookmarked by your collaborators from breaking.

When should I use ARKs compared to DOIs, Handles, PURLs, or URNs?

There are no simple answers. Identifiers (not things, but their names) are tricky to talk about, so if you hear simple answers elsewhere, beware of common fallacies.

Nothing inherent in ARKs, DOIs, Handles, PURLs, or URNs makes them more or less fit for any particular field, domain, or sector. With an identifier resolver and administrative management service, they all provide the core service of resolution (and so do properly managed URLs).

Generalizations about identifier types sometimes apply when resolution and management for that type is locked into one particular vendor or provider. For example, many PURL and Handle features and restrictions are well-defined by their respective administration silos, as are those of DOIs, which are built on top of Handles. But DOIs have metadata practices that are diverse and evolving across different DOI registration agencies.

The concrete differences that we experience, such as *metadata*, landing pages, and tool integration (eg, publishing tools), are not properties of identifier schemes per se, but properties of resolution, management, and citation services that various providers extend to or withhold from different identifier types. Those services are shaped in turn by communities of practice and by markets. Basic services are founded on a reliable database storing each identifier along with metadata elements (creator, title, date, redirection URL, etc) that describe the identified object. Extra services include link checking, duplicate detection, report generation, and searching.

From cradle to grave

When in my workflow should I create ARKs?

At object birth, or even before. We sometimes name our babies before they're born, and we name and refer to objects in the conception stages, sometimes long before they bear fruit. Depending on how elaborate the planning may be, your unborn objects could have full-function ARKs that resolve to an appropriate surrogate and return rich metadata, including persistence statements.

The only caveat is to be careful releasing (advertising) ARKs that have uncertain long-term prospects. Some identifier management systems have features to help manage and resolve unreleased identifiers (eg, EZID has a "reserved" status). The more people who know about an ARK, the harder it is to delete.

How is it that ARKs can be easy to delete?

If no one knows about an identifier but you, there's no harm in deleting or withdrawing it. Stepping back, an identifier is actually an assertion that a given string of characters is associated with specific thing. The fewer people you tell, the easier it is to scrap that assertion. If you create a URL and share it only with your closest colleagues, that is much easier to withdraw than if the URL appeared for a month on a public website, from which it was harvested by internet search engines. In contrast, it is hard to delete DOIs and Handles because once registered and made resolvable, they are effectively released to the world.

ARKs behave like URLs in this respect. Providers are free to create and share ARKs narrowly, in which case they're easy to delete.

Perhaps surprisingly, even if shared more broadly, ARKs can come with persistence statements that tell you how much or how little commitment is made to them. ARKs were designed to articulate a variety of persistence statements, but they are certainly not alone among identifiers and objects that exhibit a variety of commitment "flavors". This is why ARKs are known as high-functioning identifiers that are good at persistence rather than as "persistent identifiers".

Finally, people make mistakes. ARKs, DOIs, Handles, PURLs, and URNs are sometimes broadcast in error and need to be withdrawn. When that happens, provider best practice is to make the withdrawn identifier resolve to a "tombstone" page that explains and perhaps apologizes for the inconvenience. Despite the rumors, persistent identifiers are never guaranteed.

What is meant by ARKs supporting early object development?

People need identifiers before they know exactly what object they refer to, or if they refer to anything worth keeping. An identifier that requires mature metadata cannot be created during early development since little is known about the object. So object creators almost always initially assign identifiers that have no metadata requirements, such as URLs or ARKs.

If you start with an ARK, you benefit from being able to keep the original identifier through to public release as the metadata matures. Many objects go through intensive development and revision phases, sometimes lasting years, during which they are too immature to meet most metadata requirements. Nonetheless every object needs some sort of identifier from conception to maturity, where maturity could look like public release and further enhancement, or abandonment. It is easy to abandon ARKs that have not been released into the world.

Like the object itself, metadata *elements* need a flexible place to grow and mature over time:

- starting in the planning phase, when it just needs an *identifier*,
- at the moment of birth, when its first digital representation needs a *redirection target* URL,
- after the first analysis, when its significance and a tentative *title* emerges,
- when creating dozens of discipline-specific metadata elements that violate most metadata standards except your own,
- during post-processing by a colleague whose name you will add as an additional *creator*,
- when early feedback based on the tweeted identifier turns up a key insight and a new *contributor*,
- and so forth, through to archiving, abandonment, public release, correction, revision, enhancement, etc.

Unlike Crossref and DataCite DOIs, which require specific metadata (eg, see the DataCite schema), ARKs do not constrain any of these activities. Moreover the N2T.net resolver actually supports all of them.

If ARKs don't require it, why bother creating metadata?

Creating metadata (extra information associated with or describing an object) has several key benefits. First, no matter what the ARK redirects to – whether a landing page or a file – metadata gives users vital information about the object, such as references to newer versions, creation date, provenance, etc. For ARKs typically metadata is accessed via ARK Identifiers FAQ, version 0.91#inflections.

Metadata really eases the difficulty of working with opaque identifiers, which reveal no clues as to what they identify. In the absence of metadata you are forced to access the object itself to remind yourself what it is, and also to trust that you're accessing the correct object. Moreover, discrepancies between returned metadata and the accessed object help everyone detect identifier changes and errors.

Metadata is for grownups, and is far less important for immature objects and their identifiers than for those that have been released. Having metadata demonstrates basic provider credibility and commitment to high-functioning identifiers. Not every provider is up to this task.

It need not be expensive. Building metadata from scratch can be costly, but it's usually created and managed by object providers, in which case it can be leveraged efficiently for identifiers. Ideally, for strong persistence master metadata (maintained by object providers) should be reflected in independent systems so that it is hard for someone to tamper undetectably with identifier associations. For example, digital object repositories that obtain ARKs and DOIs from the EZID service store a copy of their metadata with EZID.cdlib.org, which in turn stores another copy with the N2T.net resolver.

What metadata is recommended for ARKs?

Metadata is messy business for all identifiers, not just ARKs. Across domains and object types there are thousands of standards, many of them overlapping yet conflicting, and each is applied according to local organizational customs and with varying levels of compliance. Choosing or creating a specification for your metadata depends on factors such as

- whether you are currently managing metadata (*hint*: stay with it unless you have a good reason to switch),
- whether you want to officially publish objects (*hint*: prepare to be able to supply author, title, date, publisher/archive, and object type),
- the requirements and capabilities of your resolver (*hint*: your IT staff or vendor might have its own requirements), and
- whether you want to store non-standard elements (*hint*: N2T allows this, but most standards and vendors don't).

Reliable cross-domain interoperation may remain out of reach, but Dublin Core, DataCite, Schema.org JSON-LD, and Dublin Kernel are common metadata specifications to consider for use with ARKs.

Why do I see ARK metadata with *who*, *what*, *when*, *where* labels?

ARKs were designed to identify anything, not just things that are, for example, publishable or purchasable. It is unnatural to model a fossil, tissue sample, vocabulary term, or Marie Curie as if each has an Author, Title, Publisher, Copyright, and Price. Instead, since 2001 an ARK typically has a four-element kernel of highly generic metadata (Dublin Kernel, inspired by Dublin Core (DC)), followed by any other metadata elements (name/value pairs) the provider wishes to provide.

Kernel metadata is structured as if in answer to the questions, who, what, when, and where regarding the expression of, or the "telling" of, an object:

- *who* "told" it (similar to DC Creator, Contributor, and Publisher, but also Inventor, Discoverer, Conductor, etc),
- *what* the "telling" was called (similar to DC Title, but also TissueSampleNumber, ArtifactBarcode, etc),
- *when* it was "told" (similar DC Date, but includes date ranges, approximate and BCE dates),
- *where* the "telling" can be found (from DC Identifier, but usually not needed because this is the ARK itself)

There's much more to say about ARK metadata, for example, applying who, what, when, and where to the content of a biography, or how an archive plans to support a dataset. More ARK metadata guidelines will be made available at arks.org. Other elements are key, such as

- *how* it was "told" (similar to ResourceType), which may dictate mappings to external metadata specs and additional elements
- redirection target URL, which is usually stored as a distinguished element of metadata
- elements holding persistence statements, to express the strength or weakness of an archival commitment

What is an ARK "inflection" and how does it differ from "content negotiation"?

An *inflection* is a change to the ending of a word to express a shift in meaning. It permits us to define a word such as "go" without also defining "goes" and "going". To an ARK that leads to an object, simply adding a '?' to the end (an example of an ARK inflection) permits us to request metadata without having to define a separate identifier for the object's metadata. This simple technique can be used by a human with a web browser. The N2T resolver supports both inflections and content negotiation.

Content negotiation for metadata is a software technique for requesting alternate formats of an object, such as the PDF or RTF form of an HTML file. Although not designed for it, historic "content negotiation" was kludged (twisted) in certain contexts to request metadata under the startling assumption that formats often used to hold metadata *are* in fact metadata and will never be objects in their own right. Unlike inflections, "content negotiation for metadata" doesn't work at all for *objects* represented in those formats (the list of which is growing and known only by private agreement), nor is it easy enough to be used directly by most human users.

Although inflections are commonly associated with ARKs, they are not "owned" by ARKs. Contrary to popular belief, identifiers don't *do* anything – it's their resolvers that *do* or *don't* support such features. So, for example, inflections and suffix passthrough are supported by n2t.net for all identifier types, but not by doi.org or handle.net for any identifier types.

Resolvers

If most ARKs run on their own resolvers, why is there also a global resolver for ARKs?

Most ARKs are created by organizations that advertise ("publish") them based at their own resolvers. For example, this ARK was published based at the gallica.bnf.fr resolver:

`https://gallica.bnf.fr/`**`ark:/12148/btv1b8449691v/f29`**

Having to run and maintain your own resolver is the cost of complete autonomy. Using your own resolver also lets you do branding via the hostname, the downside being that brands are transient and tend to make identifiers fragile. Political and even legal (eg, trademarks) pressures may make supporting older branded hostnames, hence their identifiers, difficult.

That's another reason for having the global ARK resolver. People coming across a broken identifier in the future may find its hostname no longer exists, and if it's an ARK they can extract the core identity (starting with "ark:") and present it to the global n2t.net resolver, as in

`https://n2t.net/`**`ark:/12148/btv1b8449691v/f29`**

To avoid the risk of future inconvenience, an organization – even one that runs its own resolver – may choose from the outset to advertise ("publish") its ARKs based at n2t.net.

Why does the global ARK resolver (n2t.net) *not* have the word "ARK" in it?

When demand for a global ARK resolver arose, basic principles of openness and generality prevented the designers from creating yet another silo in the DOI/Handle/PURL mold. Instead, the ARK resolver was built to be a generic, scheme-agnostic resolver called N2T (Name-to-Thing), which now resolves over 600 types of identifier, including ARKs, DOIs, Handles, PURLs, URNs, ORCIDs, ISSNs, etc. Resolution is essentially looking in a table for an identifier string, regardless of type, and redirecting it to the right place.

The same basic principles guided the design of an earlier tool called noid, which was built for ARKs but is also regularly used by organizations that mint Handles.

What do you mean by silos?

Typically, scheme-based services are designed as silos, or closed platforms, serving a particular identifier type such as Handle, DOI, or PURL. Each silo performs the same main functions – mapping names (identifiers strings) to things (objects or metadata). Excluding all but one type of identifier string may help to capture markets, but it's wasteful and non-inclusive. It requires building the same set of services over and over for each type and violates basic principles of openness.

In contrast the N2T (Name-to-Thing) resolver and EZID (identifiers made easy) management interface were designed to work with all identifiers. Effort put into any new feature can be efficiently leveraged across all types, which sometimes creates surprising flexibility. For example, ARKs are often stored in EZID with "DOI metadata", and every DOI stored in N2T can benefit from "ARK resolution features" such as inflections and suffix passthrough, which are not available via the main DOI resolver (doi.org).