

	<p>persistent identifier linking infrastructure</p>	<p>web: http://resolver.net.au/hdl/102.100.272/ON8J991QH email: policy@pilin.net.au</p>
---	--	--

Version History

Version	Date	Status & changes	Expression identifiers
V1.0	2007-12-20	Release.	PILIN/QMWL84PQH hdl:102.100.272/QMWL84PQH

PILIN Project Guidelines

Persistence of Identifiers Guidelines

To cite the *latest* version of this work use <http://resolver.net.au/hdl/102.100.272/V89DC0DQH>

To cite *this* version of this work, use <http://resolver.net.au/hdl/102.100.272/QMWL84PQH>

1 Purpose/Issue

This document gives considerations for parties seeking to ensure that the identifiers they create and maintain for things remain persistent. It presents overall guidance on how to plan for persistent identifiers in the domain. It then presents strategies for maintaining persistence when one or more aspects of the domain change: the identifier name, the identifier context, the identifier authority, the thing identified. Strategies involving changes in access relations are discussed separately. This includes cases where the thing identified moves to a new trust environment, and where the identifier moves to a new system.

2 Background

This document uses definitions and concepts in the PILIN Ontology for identifiers, summarised in [1]:

- An **identifier** is an association of a name with a thing.
- A thing is **persistent** if it is managed and maintained for a defined timespan.
- Two identifiers are **equivalent** if they both identify the same thing. Two identifiers are **synonyms** if an authority claims that they are equivalent.

Copyright © Monash University



This work is licensed under the Creative Commons Attribution-Share Alike 2.5 Australia License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/2.5/au/>

This work was created as part of the PILIN project. The PILIN project is funded by the Australian Commonwealth Department of Education, Science and Training, (DEST) under the Systemic Infrastructure Initiative (SII) as part of the Commonwealth Government's Backing Australia's Ability – An Innovation Action Plan for the Future (BAA) under the ARROW Project.

- An identifier is **preferred** according to an authority, if that authority guarantees its persistence over all other equivalent identifiers.
- An identifier is **universal** within a domain if it is unique in identifying a thing: there are no equivalent identifiers to choose from.
- An **identifier management system** is a collection of definitions, information models, policies, and data sources, used to manage identifiers.
- An identifier is **loose** if the thing it identifies can change over time. An identifier is **rigid** if the thing it identifies cannot change over time.

Persistence does not mean that the thing is maintained indefinitely, but it does guarantee that during the time span, that the thing will be maintained so that external interactions with that the thing are reliable.

Persistence is always a persistence of something, and can apply to any of the elements, qualities, or actions defined by the PILIN Ontology.

3 Scope

These guidelines apply to persistence of the following qualities, which are in scope of a single identifier management system:

- Persistence of citation
- Persistence of association
- Persistence of accountability

These guidelines do not apply to persistence of the following elements:

- Persistence of actionability
- Persistence of context

Persistence of context is covered elsewhere, in guidelines about hosting models [2], and about meaningful names of contexts [3]. Persistence of actionability is discussed in the context of separating identifiers from the service calls using those identifiers [5].

These guidelines cover the ideal case where the identifier authority has full access to information about the current status of the thing identified. In that case, the authority can anticipate all changes to the thing identified that must be reflected in the identifier. Although the strategies presented assume access to information about the thing being identified, they do not model the processes of finding out about the changes; the processes will differ depending on the situation.

These guidelines also introduce the case when access to information about either the thing or the identifier is not possible—typically because

the thing identified or the identifier is managed externally, or becomes managed externally. Keeping the identifier persistent in that case requires negotiating access to the necessary information and management functions, which in turn imposes its own strategies on identifier maintenance. Those strategies, and the circumstances that make them necessary, are described in a separate document [6], which has not been finalised as of this writing. There are also strategies for maintaining persistence during the transition of an identifier to a new access arrangement; those strategies are covered, at the conclusion of this document.

4 Guidelines: Planning for Identifiers

An identifier being persistent is not primarily a matter of the identifier remaining resolvable over a long period of time. Rather, it requires the identifier to be well-maintained over a defined period. For the identifier to be well maintained, it is important to plan for identifier maintenance: the domain in which the identifiers will be used must be modelled to a sufficient extent that changes in the domain can be dealt with without disrupting identifier functionality. This domain includes not only the identifiers, but the things identified, and the services interacting with the identifiers.

This planning motivates the strategies discussed in the remainder of the document, and we describe here the minimum components the planning should involve. As noted, this description assumes for simplicity that identifier authorities retain full access to up-to-date information on the things identified, and remain able to update the identifiers themselves. If access to either component is not there or no longer there, then the strategies need to be adjusted, to account for negotiation for access with parties outside the domain.

The basic steps in planning for identifiers are:

1. Have an information model for things
2. Incorporate identifiers in the information model
3. Decouple identifier management from information management
4. Have processes for updating identifiers when things change (automated where possible)
5. Build information management services that leverage identifiers.

The motivations for these distinct components are:

1. Work out what things in the domain *can* be identified by an identifier.
2. Work out what things in the domain *will* be identified by a persistent identifier.

3. Ensure that identifiers persist despite changes to how the things they identify are managed.
4. Ensure there are mechanisms to keep the identifiers up to date.
5. Exploit identifiers to manage the things identified robustly, despite changes to how they are managed.

4.1 Have an information model

Authorities must have a coherent and explicit model of what kinds of thing they might identify, before they start creating identifiers for them. This establishes a framework to account for any new thing to be identified, in planning both identifier maintenance and the processes interacting with things through the identifiers. The information model must account for domain user expectations of what kind of things are in the domain: identifiers must identify things that make sense to users.

If the information model changes during the lifetime of the identifier, then the identifier may point to something quite different: this will require changing the assumptions of any processes using that identifier (including external users). Processes outside the policy domain will be much harder to adjust to the new association policy than processes inside it.

Considerations on the information modelling appropriate to persistent identifiers are given in the PILIN Identifier Association Guidelines [4].

4.2 Incorporate identifiers in the information model

The information model identifies the conceptual entities in the domain that can be identified. Authorities decide separately which of those conceptual entities should have identifiers provided for them. An authority can decide to identify every entity accounted for in the information model; but this is a significant maintenance burden, and in practical terms is usually unnecessary.

Of those identifiers, some may be transient, and will require little ongoing maintenance (e.g temporary URLs, which will not last long enough to require change management). Others, as the identifier authority should determine, will be persistent: these will require significantly more maintenance, but should be fewer in number.

So there are three layers of identification in a domain: things (which may not be identified); things identified (which may be identified transiently); and things identified persistently. Considerations on the selection of things to be persistently identified are given in the PILIN Identifier Association Guidelines [4].

4.3 Decouple identifier management from information management

Information management commonly assumes that identifiers should be tightly integrated with the things they identify, so that any changes in state of the thing identified are immediately reflected in the identifier. While this guarantees prompt updating of the identifier, it also binds the identifier to the informational model realised by the current information management system. If the thing is moved to a different information management framework, or is used in a context unforeseen by the information management framework, then the persistence of the identifier can be compromised. Since identifier management is all about change management, the best guarantee of persistence is to ensure the identifier is not obsoleted by a change in information management.

So identifiers should be managed through distinct processes, loosely coupled to management of the thing identified; and it should be possible to override management of the identifier without going through management of the thing identified. (For example, if the thing and its management system go offline, it should be possible to put a copy of the thing online and associate the identifier with the copy, without having to go through the now unavailable management system.)

4.4 Have processes for updating identifiers

We propose that identifier management and information management should be decoupled; it follows that any communication between information and identifier management should be through explicit update processes, rather than assuming that update will be taken care of in the system. This is particularly necessary if the identifier and the thing identified are not maintained on the same system (which is indeed normally the case): information about the thing identified, from the data repository, will determine how the identifier is updated. The processes must deal with any differences in access between the two systems, if they are managed separately.

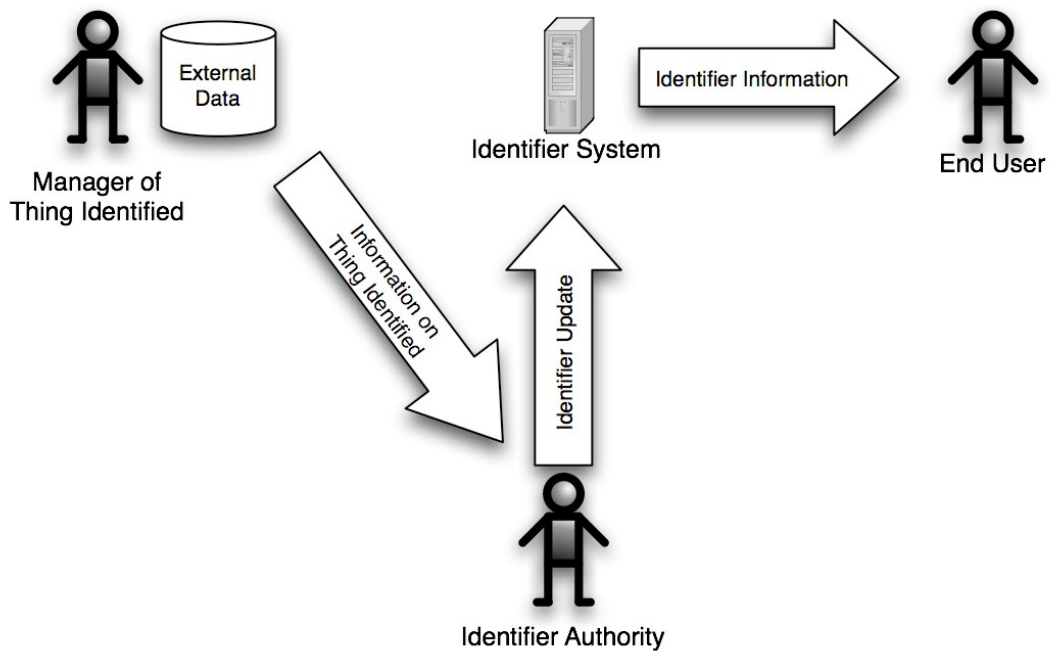


Fig. 1. Actors and processes in identifier update.

Ideally these update processes should anticipate the changes in state of the thing identified which affect the identifier (such as a change of location), and should update the identifier in tandem with the thing being changed, so that the user does not notice any disruption. For example, if a thing is moved from A to B, the best case is a workflow like:

- Thing is copied from A to B
- Identifier is updated to point to B instead of A
- Copy of thing at A is deleted.

By coordinating the timing of deletion between the two systems, the user's ability to resolve the identifier to the move object is uninterrupted.

Wherever possible, processes for updating identifiers should be automated, and not rely on human intervention. This reduces the risk that the identifier will not be updated promptly. Workflows should be in place such that a pertinent change to the thing identified automatically triggers an update in the identifier.

Such smooth and automated integration is not always possible, particularly when there are issues in arranging access to information about the thing identified. The less promptly an identifier is updated to reflect changes in the thing identified, the more the persistence of the identifier is compromised.

4.5 Build information management services that leverage identifiers

We have already noted that identifier management should be decoupled from information management, so that the identifier can be made to persist despite changes to the way the things identified are managed. Once a robust identifier infrastructure is in place, it can feed back into the way information is managed. Identifiers identify things independently of information management implementation; so they can be used as an added layer, to manage things independently of information management implementation. Using external identifiers instead of local retrieval keys to manage things makes it less likely that information management will contain processes idiosyncratic to the current implementation.

As a result, when things are migrated to a different information management system, the existing processes for managing things can themselves be migrated across with minimal disruption. If workflows are already in place to manage objects through identifiers, and those workflows do not depend on a particular information infrastructure (because the identifiers are persistent, and abstracted away from the infrastructure), then changing the infrastructure should not result in changing any of the workflows. This contributes to the overall persistence of the identifier: the less disruption during the migration of the thing identified, the likelier that the functionality of the identifier will persist throughout the migration.

5 Change Management: Name Changes (Persistence of Citation)

5.1 Why names of identifiers change

On the face of it, if the name of an identifier changes, then the identifier (as an association of a name with a thing) becomes a different identifier; so the original identifier has been destroyed, and no longer persists. However, in practice a thing may have multiple identifiers as synonyms, without one identifier superseding the other. So a more complex model is needed to explain why it becomes necessary to "change" an identifier name.

Over its lifecycle, a thing may be identified by more than one identifier. If a new identifier is associated with the same thing as an old identifier, the persistence of the old identifier is not compromised. In particular, the persistence of citation of the old identifier is not affected: any existing citations of the old identifier that have become available remain valid. This remains true even if the old identifier ceases being the preferred identifier for the thing.

However, the authority responsible for the old identifier may decide to cease maintaining the old identifier, or even to delete it. The new identifier then replaces the old identifier, rather than serving merely as an alternative for it. In that case, the extant citations of the old identifier cease to be valid: they can no longer be used reliably to gain access to or otherwise identify what they had identified. So the citation of the old identifier ceases to be persistent.

So the name of an identifier changes when an authority:

- Decides the thing shall still be identified by an identifier
- Identifies the thing with a new identifier
- Refuses to maintain the old identifier.

(If the authority decides the thing shall not be identified by any identifier, then they are already giving up on the identifier being persistent, so no change management is possible.)

A name may change either because either its context or its label has become obsolete.

The main reason a **label** becomes obsolete is it is meaningful, and the meaning of the label becomes out of date. This may be either because the thing changes, or because the associations of the label (e.g. branding) are no longer desirable. For instance:

- The identifier for a document uses its title as a label: "Persistence Guidelines". The document is retitled to "Persistence Considerations". The label is no longer in sync with the title.
- The label for a document describes its content: "Persistence Guidelines for things under control". The document is merged with another document on "things outside control". The label no longer reflects the content of the document.
- The label for a document includes mention of a sponsoring institution: "Monash726". The document becomes maintained by a different institution. The mention of the previous sponsor in the label becomes inappropriate.
- The acceptable label policy for the context changes: because a larger number of things are being identified than were anticipated, the label size goes from exactly four characters to exactly five characters.

The **context** of an identifier may become obsolete because the identifier has passed to the control of a new authority; this involves a change in access to identifier management, and is a case discussed in more detail in a separate document [6].

- E.g. An identifier is no longer maintained at Monash and starts being maintained instead at ANU.

If the identifier is still under the same authority, then the context may have become obsolete because the identifier was transferred to a new identifier management system, which defines a new operational context.

- E.g. An identifier moves from a PURL server to a Handle server in the same institution.

Such a change is usually motivated by different capabilities, community support, or policy conventions in the two systems. The old identifier ceases to be persistent because the authority decides that maintaining both identifier systems is impractical.

Any new identifier replacing an old identifier means that citations of the old identifier will cease to persist. As long as the old and the new identifier are both under the same authority, there are three strategies for addressing this: continue to maintain the old and new identifiers independently; maintain the old identifier as an alias for the new; and eliminate the out-of-date, old citations.

5.2 Independent maintenance of identifiers

The least disruptive way of addressing multiple equivalent identifiers from the user's point of view is to continue to maintain all of them as persistent, and not to cease maintaining the old identifiers. However, this makes little business sense from the identifier manager's point of view. If the old name remains valid as a citation (in both its label and its context), then there is little motivation to create a new identifier to begin with; conversely, if the new context or label are clearly preferable, there may be explicit reasons not to maintain the old context or label.

- For instance, the context uses meaningful labels which do need to change; there is a change in the acceptable format policy for labels; the old identifier management system no longer meets community requirements.

Maintaining multiple equivalent identifiers separately imposes an added resource cost, particularly if different identifier systems are involved. It also imposes an added workflow burden for ensuring the equivalents are persistently equivalent: whenever one of the equivalents is updated, workflows need to be triggered to update the others.

Maintaining independent identifiers disrupts identifier universality within the identifier domain: any thing in the domain may have more than one identifier associated with it. This does not affect the persistence of association of the identifiers, but it does make information management using identifiers more difficult. In particular, information management often uses identifiers for deduplication: systems assume and ensure that only one instance of a thing is maintained in the system. For this to happen, there must be either one identifier per thing in the domain (i.e. identifier universality), or else the system needs to keep track of all synonymous identifiers used, as an added maintenance burden (which

independent maintenance of identifiers requires). Otherwise, systems interacting with the identifiers may need to confirm explicitly that two things identified by distinct identifiers are themselves distinct.

5.3 Dependent maintenance: Aliasing

An alternative to independent maintenance of two identifiers is to alias the old name to the new. In terms of the PILIN Ontology, this means:

- the policy domain registers the new (target) name alongside the old (alias) name as equivalent identifiers, associated with the same thing;
- the synonymy of new and old name is maintained;
- to minimise administration overhead and the possibility of error, the alias is managed to be dependent on the target. This means that association data is registered explicitly only under the target name, and not registered separately under the alias.
- The alias is also actioned to be dependent on the target. That is, any actions on the alias are realised as actions on the target.
- Because the alias is both managed and actioned to be dependent on the target, the identifier management system defines the alias to be dependent on the target.

Aliasing does not require that the alias and target are in the same operational context, or even under the same identifier authority—although the latter case involves negotiating access between identifier authorities, and is one of the strategies for maintaining identifier persistence discussed in [6].

- E.g. Handle 102.100.272/persistence is aliased to Handle 102.100.272/62514
- E.g. URL <http://www.example.com/latest> is aliased to URL <http://www.example.com/docs/183.html> (as an HTTP redirect)
- E.g. Handle 102.100.272/persistence is aliased to Handle 67812/7681 (a Handle in a different namespace).
- E.g. Handle 102.100.272/persistence is aliased to PURL <http://purl.org/nla/738412>

Aliasing imposes an overhead, as the old alias is still a digital object that needs to be maintained. If the new identifier is to be the preferred identifier (which is presumably why the change happened in the first place), this imposes the added burden on the authority of actively promoting the new identifier to users. However there is no longer the added burden of coordinating separate identifier updates, and aliasing

guarantees that the old name persists in association and citation, so long as it persists in accessibility.

Aliasing, like maintaining independent identifiers, disrupts any notion of identifier universality. However, making one identifier preferred in the domain mitigates any real disruption this might cause. Moreover, the system can use aliasing information to determine whether two identifiers are equivalent straightforwardly, which makes deduplication easier.

5.4 Eliminate old citations: Patching

In some circumstances, independent identifiers and aliasing are not realistic alternatives: the identifier management system only allows one identifier for the thing identified, and the name of that identifier needs to change. This may be for technical reasons: the identifier management system does not support aliases, or uses meaningful labels which do not allow redundancy. This may also be for policy reasons: e.g. the system guarantees all its published identifiers are the preferred identifiers (i.e. identifier universality within the domain), so an identifier which is no longer preferred must be unpublished.

This means that the old identifier not only can no longer be maintained, but it should not even be cited (e.g. presented in documents): the identifier system will provide no mechanism to get from the old identifier to what it identifies, so citations of the old identifier become useless. In that case, all instances of citation of the identifier need to be *patched*: updated from the old to the new name.

Patching escapes the technical overhead of maintaining two identifiers, but it imposes a significant overhead of citation management, requiring additional resources of communication, education and policing to ensure that citations are updated. The more widely the identifier has been cited and used, the more difficult it is to patch. Once the identifier is cited and used outside the policy domain of the issuing authority, patching may well be impractical, as compliance cannot be enforced. If the policy domain is unclear or ambiguous, patching may also be impractical. If the identifier is only used within one policy domain, and policy can be enforced strongly, then patching may still be practical. But archival and historical use require that the association of the old name with the thing remain recoverable.

Example: "Leningrad" as a label for a city is changed to "St Petersburg" by the Russian Federation, after a popular vote. All instances of the label within the policy domain of the issuing authority are changed: e.g. government newscasts, atlases, rail timetables. Some instances are not within the policy domain of the issuing authority, and cannot be patched. (Old communists and disaffected youth still use "Leningrad". There is no current attempt to extend the authority's policy domain to individual speech, though there is precedent for this.) There is ambiguity over the scope of the label change (After another popular vote, "Leningrad" is still used to

refer to the surrounding administrative region [*oblast*].) Historical and archival usage remains unpatched. (History books continue to refer to the Siege of Leningrad.)

5.5 Preventing Change to Persistence of Citation

The foregoing three strategies presuppose that the name of the identifier must change, because of external factors, and cope with the resulting disruption in persistence of citation. Other strategies can be used with identifier labels to prevent any disruption to persistence of citation, or to make the transition between name changes less disruptive.

5.5.1 Avoid Meaningful Labels

Changes in meaning or context can make meaningful labels obsolete. To prevent this, meaningful labels themselves should be avoided—particularly for preferred identifiers, which are guaranteed to persist. Though labels may be presented to users in meaningful ways (e.g. as queries), the information infrastructure of a system should rely on preferred, arbitrary identifiers where available.

- E.g. A user sees on a browse screen meaningful names of object to select among, but the retrieval system for those objects uses arbitrary labels instead.

5.5.2 Use loose identifiers

The association of an identifier with a thing may be either rigid or loose: it is *rigid* if the association is only with a particular instance, but *loose* if the association is with a role a thing serves, and may be satisfied by different instances at different times.

- “John Winston Howard” is a rigid identifier; “the current prime minister of Australia” is a loose identifier.
- An identifier for a particular copy of a resource (a URL) is rigid; an identifier for a version of a resource is loose, and can be satisfied by association with any available copy of the version.
- An identifier for an abstract thing in an information hierarchy is more loose than an identifier for a concrete thing (e.g. identifier for version vs. identifier for copy); but the concepts of loose and abstract identifier are independent. An abstract thing can be identified rigidly or loosely: “version 36” is loose with respect to instances but not versions; “latest version” is loose with respect to versions. The same holds for concrete things: “the Monash library copy” is a rigid identifier for a concrete thing, but “the local copy” is a loose identifier.

Loose identifiers are generally less susceptible than rigid identifiers to disruption of their association. If the thing identified fulfils a defined role

(which users understand), then a change in instantiation for the role results in a change of the rigid identifier, but no change in the loose identifier. For that reason, citations of loose identifiers are likelier to persist longer than citations of rigid identifiers.

- “The current prime minister of Australia” continues to refer to the leader of the government despite changes in incumbency. Mentions of “John Winston Howard” as the leader of government need to be patched after 2007.

Rigid identifiers which do not refer to something in active use are also less likely to be maintained actively, whereas a loose identifier must be kept continuously up to date. So an identifier for the latest version of a document is likelier to remain persistent than an identifier for a specific version of a document—which may not be maintained after a certain timespan. Conversely, a loose identifier imposes a much greater update burden for its association data than does a rigid identifier: it needs to be updated every time the role it matches is instantiated differently, and not every time the thing identified is to be accessed differently. But those updates are explicitly accounted for in the information model for the identifier, so they can be planned for.

The preference for loose identifiers underlies the argument for using persistent identifiers instead of locators. Locators are associated rigidly with a particular file location, and persist only as long as the file stays at that location. The association data for a loose identifier, on the other hand, is not bound to any instance of the file it identifies, and indeed may not be a URL at all, but some other identifying characteristic.

There are several cases where other things are not equal, and a rigid identifier is preferable.

- If updating association data is a prohibitive burden, a loose identifier is unlikely to be maintained, and a rigid identifier is less likely to go out of date.
- Users often expect names for things to be rigid identifiers; this applies both to names of people and objects, and to the URLs users are familiar with in the digital realm. So loose identifiers (such as “latest version of”) may require educating the user as to why they are preferable.
- A loose identifier only makes sense in terms of the role it defines, which must be modelled explicitly in the information model for the domain. Identifiers for “current prime minister of Australia” and “current leader of the Labor Party of Australia” would need to be kept apart in modelling, for instance, since they do not always identify the same person.
- In some contexts loose identifiers are inappropriate, since the thing to be identified is necessarily an instance. For example, “the current prime minister of Australia” will not have supplanted “John Winston Howard” as an identifier in the medical records of his local GP.

Similarly an historical account of the Australian government will use the rigid identifier, whereas legislation will use the loose identifier, as it is supposed to hold true whoever the incumbent.

5.5.3 Do not Reuse Names within a Context

An added concern for the persistence of citation is the predictability of what the citation is associated with. If a name is disassociated with a thing in a context, and later associated with a different thing, then the persistence of citation for that identifier is destroyed: the name means something different from what it used to, and any old citations of the identifier are now misleading. Furthermore, it becomes much more difficult to recover the past association of the label, as users assume that names are not reused: if an identifier citation the user finds is resolvable, they will not think to query whether it used to resolve to a completely different thing.

To forestall the need to reuse names, authorities must ensure that the range of labels allowed in their context is large enough to cover all associations required by the project, and to safeguard against label collision. This is easiest if labels are arbitrary, reasonably large in length, and generated through a straightforward monotonic process (e.g. timestamps, sequence numbers).

5.5.4 Reuse Labels Between Contexts

Maintaining the same label across changes in context makes the label persist; this can reduce the overhead in establishing the new identifiers. Such old–new identifier pairs with the same label are called **homologues** in the PILIN Ontology. Homologues reduce the amount of change necessary between the old and new identifier, which makes the transition between identifiers easier to manage. They also preserve the meaningfulness of the labels, if the labels were meaningful to begin with. Where the context of the identifier is implicit in representations of the identifier (e.g. XYZZY instead of JACK-LIBRARY::XYZZY or JILL-LIBRARY::XYZZY), keeping the label the same eliminates the need for patching the identifier.

That said, there are constraints on whether an institution can reuse its old labels.

- The label policy of the new (concrete) context may be at odds with the old context, so that the old labels are not acceptable in the new context.
 - E.g. the new context requires URL-safe labels, which the old context did not.
- If the new context is not under the exclusive control of the institution, then there may be collisions between the old label, and

the same label already registered in the new context by a different authority.

- E.g. I am moving labels from JACK-LIBRARY to JILL-LIBRARY-CONSORTIUM, and wish to continue using the label BestBookEver to refer to the Iliad (continuing from JACK-LIBRARY::BestBookEver). But the name JILL-LIBRARY-CONSORTIUM::BestBookEver has already been allocated to identify the Mahabharata by JANE-LIBRARY, which also has access to the consortium's identifiers.
- Continuing to use the same label in the old and the new identifier creates the expectation that they are equivalent identifiers, managed by the same authority and identifying the same thing. This is exactly what the authority intends to convey; but such a state of affairs may not persist: the new identifier continues to be maintained while the old identifier does not, and the authority for the new and the old identifier may eventually be different. Coupling the new and old identifiers may ultimately be misleading, and may compromise the persistence of the identifier. Users in general cannot assume that the same label means the two identifiers are in fact equivalent: the common label may just be a coincidence. Some other mechanism (such as aliasing: see §5.3) must be consulted by users to establish this equivalence.

6 Change Management: Wrong Thing Identified (Persistence of Association)

There are circumstances in which an identifier becomes associated with a new thing. For example, a loose identifier for the latest version of a document will be updated to resolve to a new version whenever one is published. However, it is possible for an identifier to become associated with a thing not foreseen in the information model, or even something that is completely incorrect. For instance, an identifier for the latest version of a policy document might incorrectly be updated to resolve to an old version rather than the latest version of the document.

If the identifier starts identifying the wrong thing, the citation of the identifier persists (the name stays the same); but the **persistence of association** is disrupted for the identifier: the identifier no longer refers to the appropriate thing.

Association may be disrupted in an identifier for several reasons:

- The information model of the identifier has been misunderstood. For example, the identifier is used rigidly but was understood to be used loosely (the identifier points to someone filling a role, the role is filled by someone else, but the identifier has not been updated).

- The association data for the identifier is out of date, and processes to keep it up to date have failed.
- The identifier has been updated in error.

The following are strategies intended to mitigate the risk of identifier association failing. While they cannot all prevent a mistaken association being made, they can help identify errors and lead to repair, or else lessen the impact of a mistaken association.

6.1 Use typed identifiers

In general, it is difficult to enforce persistence of association automatically: it requires updates of things to be synchronised continuously with the updates of their identifiers. It is, however, easier to enforce policies which make weaker requirements of persistence, and are preconditions for persistence of association. One alternative is to type identifiers: by policy, a given identifier once created may only be associated with a particular type of thing. This imposes persistence of type of association, and persistence of functionality if the identifier is actionable. Any updates in association data for the identifier are checked against the type: if the new instance of the association violates the type constraint, the update is rejected.

- E.g. An identifier is typed as pointing to documents. Any update to its resolution data is checked against the identifier type. An authority attempts to update the identifier to resolve to a bulletin board, instead of a document. The system determines that the new resolution data does not resolve to a document, and the update is rejected.

The type of the identifier association needs to be registered, so that the identifier system can consult it, and maintained over the lifetime of the identifier. The challenge for this approach is devising an ontology of things which will persist over the lifetime of the identifier. More general types, defined through functionality rather than technology, are likelier to persist than more specific types; e.g. text rather than UTF-8, image over JPEG, document over Open Office. On the other hand, functional types alone are unlikely to provide sufficient information for services to interoperate with the thing they refer to.

6.2 Validate identifiers

An identifier management system must establish trust that its identifiers have persistence of association. To do so, the system can periodically verify that the identifier continues to be associated with the same thing. At a minimum, the system should verify that the identifier is associated with some thing, and that any resolvable identifier remains resolvable. This is a "link rot checker".

This does not guarantee that the identifier is now associated with the same thing as before. Such validation requires extra steps, including resolving to metadata and tracking any changes in the metadata resolution; and logging any changes to both the identifier registry and to the data source where the associated thing is managed.

One type of metadata used to validate the continuing association of an identifier is a message digest (“digital fingerprint”). A message digest is a representation of the thing into a string short enough to be storable as metadata in a registry, and unique enough to minimise the risk of two different things having the same digest. Validation consists of generating a message digest for the thing when it is first associated, and storing it as identifier metadata; the message digest is periodically regenerated and compared with the registered digest.

Message digests are only usable if the identifier is used rigidly, with no change to the thing associated. Any change to the content or the instantiation of the thing make the message digest obsolete.

6.3 Use redundant association data

The association of an identifier with a thing is represented through association data, which may be registered in a registry. Such data may specify how to access the thing (e.g. URLs), or how to differentiate the thing from other things (e.g. a bibliographical citation, a prose description).

Multiple kinds of association data may be used to represent the association. On the one hand, this leads to more maintenance overhead in an identifier registry: any changes to the association may force multiple updates to data; there is more data to migrate if the identifier is transferred to another identifier management system; and the data is likelier to be heterogenous, and more difficult to manage coherently.

On the other hand, multiple association data provides redundancy: even if one type of association data fails, the others may still allow the association to be recovered—so long as the resolution procedures are flexible enough to capitalise on the redundancy. The redundancy also makes it easier to detect failure while validating the association: if the association data instances for a single identifier resolve to different things, then one of the data instances is out of date. And from an archival perspective, redundancy provides more contexts of use for the thing identified, which improve reliability as well as providing evidence of how the thing identified was used over time.

6.4 Embed identifiers into things

The persistence of association between a name and a thing presupposes that both the identifier and the thing are well-managed, and that changes

in one are synchronised with the other. To decrease reliance on the management of the thing, the identifier can be embedded in the thing directly, rather than stored as metadata outside the thing. The association of the identifier with the thing can then be validated directly by inspection, once the identifier is resolved: the thing resolved to will contain the identifier name expected.

An embedded identifier for the thing can be presented to users directly, without being dependent on identifier or thing management infrastructure, even if that infrastructure is disrupted.

- E.g. if the identifier is included in the PDF, then users can become aware of the identifier even if the PDF is downloaded, and no longer accessed through an identifier-enabled repository. This remains the case even if the repository it came from is now offline.

This makes persistence possible for the identifier even when the thing is not being maintained as a digital object: the association is recoverable offline.

For a thing with textual content, embedding can be as simple as a citation of the identifier inside the document. Embedding can also take the form of digital watermarking, e.g. through steganography; validation requires a process to extract the watermarked identifier from the thing resolved to.

7 Change Management: Authority Metadata: (Persistence of Accountability)

Identifiers should be accountable to their users; this means that information on who the authority for the identifier is should be available to users. There are two related reasons for this.

The first reason is to establish trustworthiness for an identifier: if the user knows who the authority for the identifier is, they know that the particular authority vouches for the accuracy and persistence of the identifier, and given information on the authority's trustworthiness in general, they can decide whether to rely on the identifier persisting for their own processes.

The second reason is to provide fallback in case the identifier does not persist: the user can notify the authority of failure, and seek redress or restoration of functionality. In general, the user can question the authority on the identifier, making the authority accountable to the user.

In order for an identifier to be fully accountable to its users, the authority must be able to explain when and how identifier functionality was disrupted—so that they can take meaningful steps to fix any underlying problems.

If a persistent identifier is accountable, and its accountability is used to establish that the persistence of the identifier is trustworthy, then the identifier should remain accountable over the lifespan of its persistence. In other words, its accountability should itself be persistent.

7.1 Persistent Contact Details

Any identifiers used in authority metadata (e.g. institution, current manager) should themselves be persistent. Accordingly:

- The current manager of an identifier should be cited with a loose rather than a rigid identifier: by role (e.g. identifiers@example.com) rather than as a person (e.g. john.doe@example.com). For provenance (i.e. historical) metadata, rigid identifiers associated with specific individuals may be preferable.
- The current manager of an identifier may be cited with an arbitrary or arbitrary rather than a transparently meaningful identifier. Cf. the use in the Handle system of other Handles to indicate the identifier manager in the HS_ADMIN field.
- The current manager of an identifier should be identified through an identifier with persistence of actionability, or else no actionability, in preference to non-persistent actionability. For instance, it is better to use either a Handle resolvable to email, or a Handle resolvable to metadata, than an email address. An email address ties the Handle in to a specific technology, which may not persist. Having a Handle resolve to an email service allows that service to be updated later to something replacing email. And having the Handle resolve to metadata avoids the problem of tying the Handle to a messaging service in the first place.

7.2 Log identifier actions

In order to guarantee long-term accountability, the actions taking place on an identifier through an identifier management system need to be logged, and the logs should be guaranteed available for inspection for as long as the identifier itself is available.

8 Access

8.1 Establishing Access

We can define the components needed to maintain an identifier as:

- The identifier system itself

- Processes informing the identifier authority of changes to the thing identified (system inputs)
- Processes acting on the identifier to communicate information about the identifier or the thing identified (system outputs)

The system input may inform an authority to manually update identifiers based on information from the thing identified (as illustrated); but of course this can also be done through an automated system triggering updated on the identifier system. The important point to realise is that the informing process triggers the update process.

Effective identifier maintenance relies on these components all being within a trust boundary (Fig. 2).

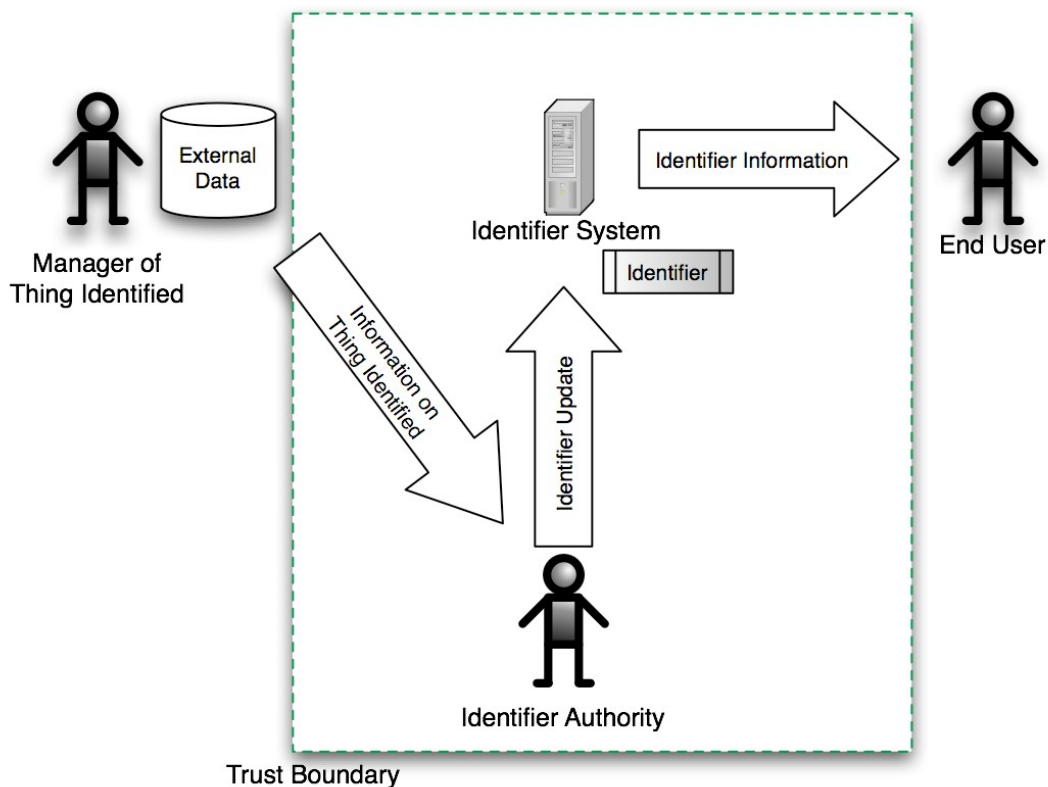


Fig. 2. Trust boundary and update processes.

The thing identified is not necessarily a component of the interactions needed for persistence. That is, management of the thing may or may not lie within the trust boundary: identifiers can persistently identify things managed externally. However, the processes triggering identifier updates (i.e. changes to the thing identified, such as its location) are an essential component of the interactions needed for persistence. The easiest way to coordinate information services about the thing identifier with update services for the identifier—making accurate updates possible—is if both services lie within the trust boundary.

We have identified two services needed within the trust boundary to keep identifiers up to date: an information service from the thing identified, and an identifier update service. If either falls outside the trust boundary—that is, if access to either is disrupted, then the identifier can no longer be kept up to date. The circumstances under which this may take place, and possible strategies for dealing with them, are discussed separately [6].

9 Guidelines: Managing Transition

Changes in access such as are described above involve a transition process between access regimes. Even if the identifier functions properly before and after transition, it may not function properly during transition, as the access regime is being reconfigured. However brief the transition process is, this means that the persistence and reliability of the identifier is still compromised.

If the transition process is managed, some steps can be taken to safeguard the persistence of association and accountability during the process. This presupposes that there is collaboration between the two domains of control involved (e.g. two institutional identifier management systems), and that both domains persist through the transition. If the transition is catastrophic (e.g. the old identifier management system goes offline without warning), then it cannot be managed.

9.1 Make lifespan of identifiers explicit

As a general principle, authorities for persistent identifiers need to establish trust even though the identifiers might one day cease to persist; and they need to allow identifier users to plan for the day when that happens. This is particularly important if an identifier context is about to expire, since such an event is typically planned for well in advance. Failure to ensure the persistence of identifiers advertised as persistent damages the trustworthiness not only of the institution, but of the persistent identifier technology and persistent identifiers themselves.

To establish that trust, a persistent identifier may expose the guarantees of persistence the institution makes, as a contract with the user. Part of the contract is the timespan during which the institution guarantees its identifiers will be maintained. Such an exposed guarantee is a feature of the ARK identifier system. Knowing the timing of any anticipated transition will help users plan for dealing with the transition.

9.2 Preserve Provenance

Data identifying the former ownership and management of the entity moving should be preserved and exposed in order to allow accountability. This applies to both things and names moving from one domain to another: this is a change in management which may affect accessibility and delivery, and which should be subject to scrutiny. The responsibility for presenting this data lies with the new home for the entity, although the presentation may originate in the original home.

9.3 Mirror during Transition

In order to preserve the actionability of the entity during the transition, the registration of the entity in a new system should precede the transfer of actionability from the old to the new system. If a thing is moving, that means that the thing should be ingested in the new system before the resolution of any identifiers changes from the old to the new. If an identifier context is moving, that means that the corresponding identifier should be registered in the new system before the old identifier is unpublished or disabled.

This means that at some stage during the transition, there will be two copies of the entity, in the old and the new domains; but at any time external users will only access one copy. Ideally the old and new systems should be in negotiation off-band, so that the old system knows to change its registration (its resolution, or the availability of its identifier) only after the new system has successfully registered its own copy. Failing that, the old system will need to somehow poll the new system, to work out when it is safe to switch across.

Mirroring has already been seen in planning for identifier persistence, as the workflow:

- Thing is copied from A to B
- Identifier is updated to point to B instead of A
- Copy of thing at A is deleted.

9.4 Advertise Move

Particularly for patching identifiers, consumers of the changed entity need to be informed promptly and effectively of the necessary change—outside of the lifespan guarantee for identifiers already suggested above. Although the patching problem is in general intractable, it can be mitigated through communication; e.g. embedding an alert into attempts to action the changing entity, so long as that embedding does not disrupt the action. Embedding suited to human consumers of actions may not be appropriate to machine consumers. A banner may be appropriate

embedding in a browser-based identifier resolver, for instance, while a warning code may be used instead in a machine-to-machine resolver.

10 References

- [1] PILIN Project 2007. *PILIN Glossary*. <http://resolver.net.au/hdl/102.100.272/HHYMV8JQH>
- [2] PILIN Project 2007. *PILIN Project Guidelines: Considerations for Ownership of Identifier Management Systems*. <http://resolver.net.au/hdl/102.100.272/461BL3DQH>
- [3] PILIN Project 2007. *PILIN Project Guidelines: Considerations for Managing Contexts*. <http://resolver.net.au/hdl/102.100.272/N8R5K6DQH>
- [4] PILIN Project 2007. *PILIN Project Guidelines: Identifier Association Guidelines*. <http://resolver.net.au/hdl/102.100.272/WBNMH9DQH>
- [5] PILIN Project, 2007. *PILIN Project Guidelines: Identifier Service Guidelines*. <http://resolver.net.au/hdl/102.100.272/1KKBLPDQH>
- [6] PILIN Project, forthcoming. *PILIN Project Guidelines: Identifier Persistence under Changed Access Conditions*.

Copyright © Monash University



This work is licensed under the Creative Commons Attribution-Share Alike 2.5 Australia License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/2.5/au/>

This work was created as part of the PILIN project. The PILIN project is funded by the Australian Commonwealth Department of Education, Science and Training, (DEST) under the Systemic Infrastructure Initiative (SII) as part of the Commonwealth Government's Backing Australia's Ability – An Innovation Action Plan for the Future (BAA) under the ARROW Project.