

**Version History**

Version	Date	Status & changes	Expression identifiers
V1.0	2007-12-11	Release	PILIN/ 1TZWM4PQH hdl:102.100.272/1TZWM4PQH

# PILIN Project Guidelines

## Meaningfulness of Labels in Identifiers

To cite the *latest* version of this work use <http://resolver.net.au/hdl/102.100.272/D6N8F0DQH>  
To cite *this* version of this work, use <http://resolver.net.au/hdl/102.100.272/1TZWM4PQH>

### 1 Purpose/Issue

This policy describes issues confronting projects in determining a label policy. It addresses in particular the benefits and costs for projects choosing between meaningful, opaque, and arbitrary labels.

### 2 Background

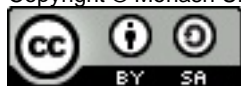
This document uses the following definitions and concepts from the PILIN abstract identifier model [1]:

- An *identifier* is an association of a *name* and a *thing*.
- A *name* is a pair of a *label*, and a *context* within which the label is unique.
- Names are *encoded* for presentation, to fulfil specific purposes.
- Names can be parameters of *services* acting on the identifier.
- Service *calls* (including their name parameters) have their own encodings, which may be distinct from the usual encoding of names in isolation.

This document addresses the question of whether names should be meaningful or not. It discusses:

- Whether a name should have *meaning*.
- If a name has a meaning, whether the meaning should be *transparent* (obvious) or *opaque* (hidden) or *obfuscated* (partially hidden).

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 is created as part of the PILIN – Persistent Identifier Linking Infrastructure project. The PILIN project is sponsored by the Australian Commonwealth Department of Education, Science and Training under the ARROW Project.

- If a name has meaning, should the meaning be deemed *relevant* to the thing identified.

### 3 Scope

This document does not address all issues involved in determining a naming policy. It only deals with issues specific to the labels within a name.

Names consist of labels and contexts. Contexts themselves have names. Names of contexts have their own particular issues, which are addressed in a separate context guideline document [2].

Other guideline documents address:

- Constraints on form and format of names to guarantee identifiers are citable [3].
- Reusing labels within a context [4].
- Ways of encoding and citing names [5].
- Meaningful versus opaque labels and encodings for contexts [2].

### 4 Guidelines

#### 4.1 Meaningful and Arbitrary Names

A name is *meaningful*, if there is a direct relation between the name and some relevant fact about the thing being identified. If the fact is the value of some attribute of the thing identified (such as title, location, or creation date), then using that value as a name makes the name meaningful.

**Example:** The name 20070826 is directly related to the fact “the date of creation is August 26, 2007”, and that fact can be inferred from the name just by looking at it (inspection). This name is meaningful if the date is some attribute of the thing being identified (e.g. its creation date).

A name is *arbitrary*, if there is no relation between the label and any relevant facts about the thing being identified.

*Should my names be arbitrary or meaningful?*

<i>Arbitrary Name: Pro</i>	<i>Arbitrary Name: Con</i>
Unaffected by any changes in thing identified: strong guarantee of	Often hard for human users to remember and express.

persistence.	
Can be easy to generate name for thing with minimum of management overhead. (e.g. timestamp)	Imposes requirement of registry of metadata about the thing, instead of allowing retrieval of metadata by inspection of the name.
	Requires label generating process to guarantee arbitrariness.
	Any meaning in the identifier, even if it is not relevant to the thing identified, may still be exploited in unpredictable ways, e.g. as a security risk.

## 4.2 Transparent, Opaque, and Obfuscated Meanings

A meaningful name is (*semantically*) *transparent* when the fact it relates to can be inferred by inspection. For example, the timestamp 20070826.

A name may instead be *opaque*: there *may* be a correspondence between the label and a relevant fact about the thing being identified, but that correspondence cannot easily be inferred by inspection.

Example: A name is a timestamp (20070826), which is reversed (62802007) and encoded in Base 36 (11E2C7). The name is meaningful but opaque: a user needs to know about the obfuscation process to work out what time the name corresponds to.

A name is *obfuscated* if there is a correspondence between a name and a fact about the thing being identified, but the fact cannot be inferred from the name. The name is predictable given the fact — but only to those who know about the fact.

**Example:** A name is a numeric OID (e.g. "1.4"). The first number in the OID represents an experimental subject. The second number represents , experiments performed on those subjects. The correspondence of names and subjects is semantically transparent: we can predict that "1.5" and "1.7" identify different studies performed on the same subject. But the mapping of those numerical labels to particular subjects is obfuscated: we don't know whether the "1" in "1.7" is John Doe or Jane Smith.

*If meaningful, should my identifiers be semantically transparent, or obfuscated?*

<i>Semantically Transparent Identifier: Pro</i>	<i>Semantically Transparent Identifier: Con</i>
Easy to generate name for thing with	Vulnerable to changes in thing

minimum of management overhead.	identified: name can become semantically out of sync with the thing, compromising its persistence.
Easy to remember for human users.	If names are kept in sync with the thing, then old identifiers for the thing need to be replaced where-ever they have been used. This "identifier patching problem" increases the more widely the old identifier was used.
May allow retrieval of crucial metadata about the thing by inspection. (e.g. name, location, relations)	

<i>Obfuscated Identifier: Pro</i>	<i>Obfuscated Identifier: Con</i>
Some retrieval of crucial metadata about the thing may still be possible by reversing the obfuscation. (e.g. unscrambling a timestamp).	Requires label obfuscating process during generation, to guarantee arbitrariness.
Some retrieval of crucial metadata about the thing may still be possible by inspection. (e.g. "1.4" is related to "1.5")	If identifier is widely known to be obfuscated, users may still expect that the name remains in sync with the thing, as long as they can recover meaning from the name.
Reduces vulnerability to changes in thing identified: the name can become semantically out of sync with the thing, without users expecting the name to be brought back in sync.	May be hard to remember and type for human users, depending on obfuscation algorithm. (e.g. hashes)
May be easy to remember and type for human users, depending on obfuscation algorithm. (e.g. small numbers)	

### 4.3 Relevant by Policy

Even if a name is meaningful, policy may dictate that the fact encoded in the name should be treated as irrelevant. This means that the name can again be considered arbitrary. For instance: a name is initially meaningful as an identifier: the thing's title is used as the name. But by policy the name is treated as arbitrary: no attempt is made to interpret the name, and the name is not updated even if the thing's title changes.

**Example:** Unicode names of characters are identifiers which remain the same by policy—even if it is later discovered that they are incorrect as

descriptions of the character. The name is relevant (since it describes the character with respect to a script), but it is treated as arbitrary by policy.

*I want an arbitrary identifier, but have to use meaningful identifiers instead (e.g. for legacy reasons). Can I treat these identifiers as arbitrary by policy?*

<i>Meaningful Identifier Treated as Arbitrary: Pro</i>	<i>Meaningful Identifier Treated as Arbitrary: Con</i>
Easy to generate name for thing with minimum of management overhead.	Imposes communications overhead: users should not to make assumption of meaning in the identifier—despite what the identifier looks like.
Easy to remember for human users.	Foregoes most advantages meaningful identifiers have over arbitrary identifiers: users can no longer assume semantic transparency.
Semantically transparent identifiers are no longer compromised by changes in the thing identified: the identifier persists even if the label no longer reflects the thing as it used to.	Requires registry for retrieval of metadata about the thing, instead of retrieval by inspection of the name: inspection can no longer be trusted.

## 5 References

- [1] *PILIN Glossary*,  
<http://resolver.net.au/hdl/102.100.272/HHYMV8JQH>  
 hdl: 102.100.272/HHYMV8JQH
  
- [2] *PILIN Project Guidelines: Considerations for Managing Contexts*.  
<http://resolver.net.au/hdl/102.100.272/N8R5K6DQH>  
 hdl: 102.100.272/N8R5K6DQH
  
- [3] *PILIN Project Guidelines: Form of Labels*,  
<http://resolver.net.au/hdl/102.100.272/0HJ9X8JQH>  
 hdl: 102.100.272/0HJ9X8JQH
  
- [4] *PILIN Project Guidelines: Persistence of Identifiers Guidelines*,  
<http://resolver.net.au/hdl/102.100.272/V89DC0DQH>  
 hdl: 102.100.272/V89DC0DQH
  
- [5] *PILIN Project Guidelines: Identifier Service Guidelines*,  
<http://resolver.net.au/hdl/102.100.272/1KKBLPDQH>  
 hdl: 102.100.272/1KKBLPDQH