NicNames application

Draft 0.1.1, 30 Jan 2009 – Thomas Rutter – Please comment

Contents

Aims

- Provide a unique identifier which can be used to refer to a particular person in institutional repositories and other external applications which do not otherwise maintain identity records.
- Provide a tool to help distinguish the identity and retrieve the correct unique identifier of a person given their name and a small amount of knowledge about them, even if that person's name is ambiguous or multiple people share the name. Such a tool should be easily integrated into external applications where data entry is done.

Without NicNames, repositories and other external applications which do not otherwise maintain identity records suffer from the following problems:

- lack of a way to reliably retrieve or group records by a certain author, if the author's name is different across different publications.
- lack of a way to distinguish between two authors with the same name.

Solving these problems in institutional repository software unable to maintain identity records requires deciding upon one unique, authoritative form of each author's name, and using that same form throughout the repository. It has been decided that this approach is not satisfactory for an institutional repository, because:

- There are legitimate reasons why an author may be known by different names in different contexts within the same repository, including situations where this would be more correct and useful than denoting one name to be the 'authoritative' form.
- In the unlikely event that two authors share the same name, it would be more correct and useful to allow both to be referred to by this name in the repository, than to modify one or the other of their names to ensure uniqueness (by adding middle initials or other information to the end of the name that they wouldn't otherwise use).

Data model

NicNames unique identifier

The NicNames database shall give each identity record its own unique identifier. This will allow external applications, such as repositories, to associate their own records with an identifier from NicNames.

The identifier shall be a positive number, incrementing in sequence – thereby containing no confidential, private information or information which will go out of date. The numeric identifier shall be unique only to that particular installation of NicNames. This ensures that anybody is free to set up their own installation of NicNames, should they choose to, without needing to cooperate with any other organisation or central authority when adding new records.

Identities shall also have a mechanism to be associated with the same identity in other NicNames databases, just as they might be associated with the same identity in external databases such as Scopus, Researcher ID or People Australia.

Where-ever an application might refer to an identifier from an external NicNames database, a prefix should be used with the identifier, which indicates the NicNames database that the identifier came from. Where NicNames exposes its data to other applications via a web service, this shall include its suggested prefix.

It would be advantageous for prefixes used to describe a given NicNames database to be consistent among other databases. Therefore, some global and exclusive way of choosing a prefix could be used. A central governing body which maintains a list of the prefixes for NicNames installations could be used, however, this would require ongoing maintenance, and it would also impact upon the freedom for anybody to set up a NicNames installation of their own, if they choose to, without needing approval or cooperation from others. Therefore, a prefix based on the domain name of the institution, such as “swinburne.edu.au” may be an appropriate convention, as it is globally unique and would need no governing body beyond obtaining a domain name, which an organisation is likely to already have. Institutions could run more than one installation and give prefixes containing additional information, like department.swinburne.edu.au. Note that a NicNames unique identifier is not intended as a URL or resolver service which can locate a NicNames installation; just an identifier that can associate a record in one system with a record in a known NicNames installation.

Any organisation who chooses to install NicNames can, at their option, choose to share their identifiers with anybody else or publicly advertise the availability of open web services for accessing their NicNames data including their identifiers. It is expected that institutions will share their data with partner institutions.

Properties associated with an identity

Identities in NicNames can have any number of properties associated with them. This includes one or more names or forms of names, resources, other people, identifiers in external applications and more. The additional properties can aid in distinguishing the identity and retrieving the correct unique identifier for a person even if the name is ambiguous or the same name is used by multiple identities.

All properties associated with a name are optional and non-exclusive other than the NicNames unique identifier. That is, a record may have zero or any number of each type of property. The smallest record which could possibly be useful would have, at a minimum, only one name or other property in addition to the NicNames unique identifier.

Each identity may have:

- Names
- Related resources
- Related events
- Related organisations
- Related people
- Related identifiers
- An optional comment

Names

Each person may have any number of names. Sometimes, we can distinguish between types of names, and variants of the same name.

Examples:

- Person has 'publishedasname' of surname 'Smith', givennames 'Michael J.'
- Person has 'employeename' of no surname, givennames 'Madonna' on dates 1984-
- Person has 'publishedasname' of surname 'Kim' and givennames 'Il-sung' and surname comes first, and this is a 'romanised' version of name B (which is a 'publishedname' of surname '김' and givennames '일성').

A name associated with a person may include:

- surname
- givennames
- title
- surnamefirst (yes or no)
- type (see below)
- pointer (ie self-referential foreign key) to name it was derived from
- derivetype (see below)
- comment

- (statistical information for internal use)

Types of names:
- publishedasname
- birthname
- legalname
- employeename
- nickname
- other

Types of ways one name can be derived from another
- shortened
- romanised
- alternatespelling
- alternateorder
- lengthened
- other

Resources

Each person may be related to a number of resources such as journal articles, journals, and more. Their role ie 'author' or 'contributor' may be specified.

Examples:
- Person 'contributed' to a 'journal' called 'Planets monthly'
- Person 'authored' a 'journalarticle' called 'Why planets rotate' in 1998
- Person was 'citedin' an 'essay' called 'Impact of green vegetables'

A resource associated with a person may include:
- title
- resourcetype (see below)
- reltype (see below)
- startdate
- enddate
- comment
- (statistical information for internal use)

Types of resources:
- book
- bookchapter
- conferencepaper
- journal
- journalarticle
- technicalreport
- thesis
- newspaperarticle
- workingpaper

- essay
- speech
- other

Types of relations between person and resource:

- authored
- edited
- contributed
- citedin
- other

Events

Each person may be related to a number of events, such as conferences, meetings, ceremonies and more. Their role at that event eg speaker, organiser, attendee may be specified.

Examples:

- Person was 'attendee' of 'conference' called 'Astrorama' in 1998
- Person was a 'speaker' at 'ceremony' called 'Installation of new chancellor' in 2008

An event associated with a person may include:

- title
- eventtype (see below)
- reltype (see below)
- startdate
- enddate
- comment
- (statistical information for internal use)

Types of events:

- conference
- meeting
- social
- ceremony
- other

Types of relations between person and event:

- attendee
- awardee
- speaker
- organiser
- other

Organisations

A person may be related to any number of organisations, where the organisations may be institutions, companies, publishers, or more.

Examples:
- Person is associated with 'employer' 'institution' called 'Swinburne University of Technology' in 2008
- Person is associated with 'researchcenter' 'institution' called 'Centre for Online Technology'

An organisation associated with a person may include:
- title
- orgtype (see below)
- reltype (see below)
- startdate
- enddate
- comment
- (statistical information for internal use)

Types of organisations:
- institution
- company
- association
- publisher
- other

Types of relations between person and organisation:
- employer
- publisher
- researchcentre
- member
- owner
- subscriber
- associate
- other

Relations between people

People may be related to each other. Any person may be related to any number of others, for example as co-authors, colleagues, siblings and more. Types are two way, and in some cases the directions matters ie parentof indicates that the former is a 'parent of' the latter.

Examples:
- Person was 'coauthor' of Person B in 2004
- Person was 'employerof' Person C

A relationship associated with two people may include:
- type (see below)

- startdate
- enddate
- comment
- (statistical information for internal use)

Types of relations between two people:
- coauthor
- colleague
- sibling
- parentof
- employerof
- other

External identifiers

People will undoubtedly have unique identifiers with different organisations which can help us tell them apart such as scopus id, a local staff id or researcher id. They may also have identifiers which they have created themselves such as websites, openid or email addresses.

External identifiers represent information about a person's identity in a given context just as a name does, though external identifiers are treated as a single string – without the additional logic for separating it into surname, given names, title and more.

Examples:
- Person has 'emailaddress' of 'tony@example.com'
- Person has 'localstaffid' of 'A099288'
- Person has 'scopusid' of '14057825000'

An identifier associated with a person may include:
- identifier (the identifier itself)
- idtype (see below)
- startdate
- enddate
- comment
- (statistical information for internal use)

Types of identifiers:
- localstaffid
- localresearcherid
- emailaddress
- openid
- website
- scopusid
- thomsonresearcherid
- peopleaustraliapid
- othernicnamesuid

- other

Database format

The database shall be MySQL 5.0 (or higher).

Text stored in the database will use the UTF-8 character encoding, an efficient way of representing any character (letter or punctuation) from any language, including accented and non-Roman characters.

This does not mean that it will be necessary to place all information into the NicNames database in its original language and alphabet. Translations to English and anglicised forms of words are welcome, particularly when the information is known in the same form in other places such as the institutional repository.

Information containing right-to-left text or text requiring complex ligature rules such as Arabic may be stored, though is not likely to be displayed correctly in the online user interface or other applications accessing the NicNames data.

The NicNames application, which is written in PHP, will make use of a library providing functions for operating on UTF-8 strings, to ensure that the application is as portable as possible.

Index

The NicNames database maintains an index for fast per-word searching. There shall be no lower or upper limit on word length, ensuring that all words, including single-letter initials, are searchable. Each word will be stored along with two phoenetic representations, one forwards and one reverse based on an algorithm similar to soundex or metaphone.

The index will be built on a per-record basis when a record is created or modified.
This should allow for query times in the tens of milliseconds when searching based on words found in names, related people, publications, identifiers, organisations, etc.

After the index is consulted, a large list of possible matches is found. The next step is to sort these by relevance using a more sophisticated algorithm which can take context and importance of the different fields into account.

User interface

NicNames would have its own online user interface. This interface would allow the user to view and modify all of the information that is recorded for an identity. Modifications may be done by:

- Adding new information to an identity record, such as a new name, paper or co-author
- Modifying or deleting existing information on an identity record
- Creating a new identity record or deleting an existing one
- A 'merge' feature allowing two identities to merge, in the event that the user has discovered

that they are indeed the same person.
- A 'split' feature allowing one identity to split in two, in the event that the user has discovered that some information associated with an identity actually refers to a different person.

Search query

Using the online user interface, it will be possible for a user to look up records by typing in names or other information, including partial or similarly spelled words, related to an identity.

In response, a number of matches will be shown. These will be sorted according to an algorithm which can take into account:

- The similarity in spelling between each typed in word and the matched word.
- The degree to which the typed in words match a full field. For example, if the two words typed in match one identity's first name and surname, this will be given higher ranking than if one of the words matches part of an identity's name and another word matches one of their co-authors or associated organisations.
- Where in the identity record the word was matched. For example, words matching part of a name should be given higher ranking than words matching within a person's publications or co-authors.

Displaying a record

Once the user has located a specific identity record in the user interface, they will view the main screen for that record. The screen will list all information associated with that record, which may be as little as a single name and an external identifier, or may be much longer, including publications, publishers, co-authors, and more associated with the identity. An edit link will be given next to each field, so that it may be edited or deleted on that page.

Names associated with the record will be displayed first, and most prominently. Other information will be grouped.

Most information may contain an additional comment as further information. If present, this will be displayed alongside the piece of information it applies to. For instance, a comment relating to a particular name will be displayed below that name in smaller text. Comments shall be treated as important to the user – if the user entered a comment it is because any subsequent person viewing that record should see that comment.

Access to the user interface

A record is likely to contain information that is sensitive in nature and must be kept private, but is nonetheless a useful or necessary way of deciding on matches. This may include internal staff IDs or staff email addresses, which are needed for matching between the database and imports of data from an HR database. The user interface should therefore be made available only to authorised people within the home organisation.

When data is exposed to other applications via web services, the same shall not apply. Private and confidential information will not be shared through external interfaces.

Merge feature

To perform a merge, the user will first need to locate the identity record of the first identity to merge. The user will then be able to press a merge button or select merge from a menu, where they will be prompted to locate the record for the second identity to merge. Once the user has confirmed that all information is correct, the merge will be performed.

All of the information from both identities shall be combined to form the new one. If a piece of information for both is identical, it will be combined. For example, if both identities had the name “Harry Smith” associated with them, that will be combined into one. However, if one had the name “Harry Smith” and other other had a name “Smith, Harry S.”, then the new identity would have both names associated with it. The same goes for other information such as co-authors, publications, and more. Of course, once merged the user will then be viewing the resulting record, and will be able to do further modifications if needed.

The resulting record will use the NicNames internal unique ID of the first of the two merged identities only. The other one shall be added into the record as additional information.

Split feature

Splitting can be performed if the user discovers that information related to two separate people exists in a single identity record within NicNames. Splitting will separate selected information into a new identity record.

To perform a split, the user will first need to locate the identity record containing the information they wish to split into a separate identity. The user will then be able to press a split button or select split from a menu, where they will be prompted to select which pieces of information should be removed from this identity and placed into a new identity. For example, the user would be able to select any number of names, associated people, organisations or external identifiers that belong to the alternate identity to be separated into a new identity.

After splitting, the user will be shown the record of the newly created identity, and will be able to further modify it. The new identity will be allocated a new NicNames internal unique ID. The source record will continue to have the ID it had previously.

Query service

The ability to search for or retrieve records in the database shall also be exposed through a web service, allowing for search queries or requests for bulk records to be conducted from other applications over the network. Existing applications may be modified to access this service by way of a plugin.

It shall be possible for such a query to be made by Javascript over HTTP, to keep modifications to the external application as simple as possible. Doing a query in such a plugin could involve the end user typing words from a name, seeing a few matches to choose from, and either choosing one or declaring that there is no existing match (and therefore that the record should be considered a new identity). The web service from NicNames will return its unique identifiers and other information for each match found, such that the external application can choose a match and then use the unique identifier to associate one if its records with an identity from the NicNames database. In a simple plugin, this identifier could be added as a hidden form field to the external application. The user would not need to know this unique identifier; it would be included automatically once they submit the information.

This service is publicly accessible, and read-only. It does not cause any changes in the NicNames database. The only communication that NicNames will receive from the external interface would be the search terms or query entered. In return, information from the NicNames records from close matching identities will be returned. Such information will contain all information for the matched identities except that which must be suppressed due to privacy concerns – such as contact details and staff IDs.

The query service may be used to enhance a submission process for an external application that's available to the public – such as a self-submission service. The publicly available nature of the service would mean that anybody could build an external tool that uses this functionality, without needing cooperation from the owner of the NicNames installation. The owner of the NicNames installation could, however, disable this service or restrict its use if they chose to do so.

An external application may conduct more than one type of query using this service:

- A query based on information found in an identity. The results returned are the same as for a search query done locally in the NicNames application. Results are limited to near matches only, ordered by relevance.
- A query for grabbing records in bulk. This allows other applications to do bulk harvesting or 'synchronisation' of data within the NicNames database. The external application may supply a minimum last modified date and a maximum number of results for the query. All records returned by NicNames shall be order by last modified date. This shall allow for external applications to do incremental updates, fetching only records which have changed since their last query. It will also allow for external applications to resume the last recorded downloaded after fetching incomplete results.

Data format

The query service shall return results in a data format that is easy to parse and translate into other formats, and is capable of expressing as much of the information and relationships tracked by NicNames as possible. The data format may be a new format specifically for NicNames, or it may be an existing format capable of representing identities and their names, relationships with other people, things, organisations and identifiers.

Bulk import or 'harvesting'

The NicNames application shall be capable of importing many records at once from certain sources

– for example an institutional HR database or a repository.

The operation shall be safe enough that it can be done regularly, such as every night, without human intervention every time. Many such imports, particularly from sources such as an HR database, will involve re-importing a lot of identities that are already in the system. Automatic processing of these records will be straightforward.

Where new records exist that haven't been imported before in their current form, the system will need to make a decision as to what to do with them.

- Does the new information relate to an existing identity in the database? If so, which one?
- Does the new information relate to a new identity that is not in the database?

There will be a process whereby if any information appeared new or changed it is flagged for review, so that a human reviewer can log in at a later date and inspect the new information and make sure it has been associated with the correct identity. In most cases it is hoped that the system will have made the right decision and nothing will need to be corrected. However, in some cases the decision may need to be overridden.

- New information may have been added to an existing identity when it should have created a new identity. This may be the case where a new, previously unknown person has a very similar name to someone already in the database, and other information about them also implies that they are the same person. A user will use the 'split' function of the online interface to split the new information into a new identity.
- New information may have been added to one existing identity when it should have been added to a different one. This is likely to be an exceptionally rare occurrence and may indicate a fault or unusual change in the source database being used. Usually, a bulk import would only be done from a source that maintains its own unique identifier for a name. If the unique identifier changes in the source, and at the same time the person's name and other information changes to resemble another user in the NicNames database, new information may be added to the wrong NicNames identity record. The user could use a 'split' function in the user interface to remove some of the information into a new record, followed by a 'join' with the other record. Alternatively, the user could manually remove the information from one identity and re-enter for the other.
- A new identity may have been created in the database for the new information, even if the new information should relate to an existing identity record. The circumstances which may cause this are similar to the above – the unique identifier provided by the source database changes at the same time as the name or other information changes significantly enough that based on an automated comparison it can no longer be judged to be the same person. The user would use the 'join' function in the user interface to correct this.

Persistence of user corrected information

Any manual corrections made to data that has been imported will continue to be preserved after subsequent imports of data. That is, once the user has told the NicNames database which information belongs to which identity, subsequent imports containing the same information will continue to be associated with the identity specified by the user. A user should not have to make the same corrections twice.

If the system is in doubt, it will trust information that has been associated with an identity manually (by a user) more than it will trust information that has been automatically associated with an identity.

Idempotence

Performing the same import two or more times should have the same effect as importing it only once. Therefore if an import generates a mistake, then the mistake will not worsen (or change at all) if the same import is made again, and again. Thus, it will be safe to run unattended imports of the same data with high frequency, as frequency of updates will not worsen problems.

Data formats supported

In order to support imports from a given service, NicNames would need to support the protocol used (at this stage, this should include HTTP and FTP at minimum) and it would need a stylesheet such as an XSLT stylesheet in order to translate the data from the the given source into a data format for NicNames to use internally. The use of stylesheets such as XSLT should make it easier for users of NicNames to adapt it to other data formats, especially those based on XML.