Problems with the current "speling.org" system

Jacob Sparre Andersen

22nd May 2005

Abstract

We out-line some of the problems with the current "speling.org" system, as well as some ideas for resolving the problems.

1 Introduction to the "speling.org" system

The "speling.org" system is a collection of tools for creating dictionaries. The primary task of the system has so far been to manage the production of the free Danish, Faroese and Swedish dictionaries for spell-checking.

The system is based on a principle of not throwing information away. The source code for a dictionary is thus a log of all the proof-reading messages the proof-readers (including the editors) have entered. The information in this log is then processed by (basically) counting the number of entries ("votes") for and against each piece of information for the dictionary, and the information with sufficiently more positive than negative entries is included in the compiled dictionary.

2 Problems

- The system is based on *spellings* and not on *words*. This leads to lots of redundant – and thus possibly inconsistent – information as soon as we are interested in more than just simple spell-checking.
- It takes a long time to compile a dictionary. We should consider structuring the source code and/or tools in a way that gives shorter compile times.

- We haven't got a sensible way for proof-readers to correct other errors in the dictionary than incorrect spellings.

3 Proof-reading tasks

- Add, approve, reject or correct data[1].
- Clone a modified version from a *word*.
- Collect two words into one.
- ...

4 Data needed for a dictionary

To create a dictionary for a language, we need to collect different kinds of information. Here we structure the required information in terms of creating a *word*-based dictionary system.

Basic information:

- List of word classes.
- List of topic categories. – Possibly with information about which categories are subsets of which.
- A list of conjugations for each word class.

Information about *spellings*:

- Frequencies of the *spelling* in specified text corpora. (zero or more instances)
- References to *words* the *spelling* is a form of. (zero or more instances)
- ...

The list of *spellings* can be used to keep track of missing *words*/conjugations in the dictionary, and thus to find inconsistent data in the system, which should be sent to proof-reading.

[1] On a basic data format level this should be reduced to *approving* or *rejecting* – possibly new – data.

Information about *words*:

- Reference spelling. (exactly one instance)
- Word class. (exactly one instance)
- Meaning/description. (exactly one instance)
- Correct spellings (zero or more instances) for each of the conjugations of the specified word class.
- Common misspellings (zero or more instances) for each of the conjugations of the specified word class.
- Translations (links to words in other dictionaries). (zero or more instances)
- Synonyms (links to other words in the dictionary). (zero or more instances)
- Antonyms (links to other words in the dictionary). (zero or more instances)
- Composition (links to the two words in the dictionary the word is composed of). (zero or one instance)

Information about *spellings* of *words*:

- Hyphenation (marking where the spelling can be hyphenated).
- Pronunciation. – Include regional differences? Or should that go in different dictionaries? – A phonetic writing has to be selected.
- Pronunciation examples (links to sound files).
- ...

Links to *words* should be made unique. One way to do this is to generate a pseudo-random number (for example using a MD5 hash function) out of the first registered reference spelling, class and description of the word. But it is just as efficient to use a simple sequentially allocated number together with a reference spelling of the word.

The source code for the dictionary should not just consist of this information, but also of who has approved/rejected each bit of information.

5 Notes

Problems with the current system:

- Spelling-based, not word-based
- Text-based (slow)
- Lacks a practical way of proof-reading other data than spellings

Proof-reading tasks:

- approve/correct/reject information
- clone a word - with changes
- collect two words as one
- ...

Information about spellings:

- frequencies in corpora
- zero or more words it is a spelling of (zero ⇒ not correct)

Information about words:

- word class (one)
- meaning/description (one)
- correct spellings - zero or more for each of the conjugations of the word class
- frequent misspellings - zero or more for each of the conjugations of the word class
- translations - zero or more references to words in other dictionaries
- synonyms - zero or more references to other words in this dictionary
- antonyms - zero or more references to other words in this dictionary

5.1 Indirect data about a word

- Conjugated following *pattern*
- Conjugated like *word* (shall be a part of a chain ending in a pattern to have any effect)

5.2 New format

Warning: This does not handle information about *spellings* and *spellings of words*.

Every record shall contain the following information:

- Word identifier.
- Proof-reader identifier.
- Source identifier (can be an authority).
- Information type ("conjugated as", "belongs to the word class", ...)
- The actual information string.
- Approved/rejected.

The records can then be sorted out in groups (files) based on the word identifier.

For each word one report should be generated with the favoured (most likely correct) information for publication. And another one with all the variations of the actual information for proof-reading purposes.

5.3 Updated authorities

The official definition of Danish is in the book "Dansk Sprognævns Retskrivningsordbog" (often just denoted by "RO" or "RO*publication year*"). This book is occasionally updated. As a side-effect our selection of which words belong in the main dictionary changes.

Some steps to take to handle this:

- Introduce categories corresponding to each version of "RO" as well as a plain "RO" category.
- A newer version of "RO" will always count as a higher authority than an older version for the plain "RO" category.

- For a specific "RO*year*" category the distance to the year decides the ranking of the different "RO" versions as authorities.

- If a word is found in the *year*-version of "RO", it should have that version as an authority approving that the word belongs in the categories "RO" and "RO*year*".

- If a word isn't found in the *year*-version of "RO", it should have that version as an authority rejecting that the word belongs in the categories "RO" and "RO*year*".

Using this strategy, it should be possible to generate dictionaries equivalent to the various versions of "Dansk Sprognævns Retskrivningsordbog" by choosing the appropriate "RO*year*" categories[2].

A side-effect of this is that some words may be in the situation that they are without an approved category. The proper handling of these words is to consider them incorrect, rather than dumping them in a general "common words" category.

Due to the different definition of category handling in the log/ds format of the current version of speling.org, all approving records in the current format data should be considered containg approval of the word belonging to at least one category.

5.4 Special conjugations

For Germanic languages it is useful to consider the forms of a word which can be used as the first half of a composite word as a special kind of conjugations. But the system has to be able to make word lists, where these strings aren't included as ordinary words.

I am not sure if this should be done in the post-processing step or if it should be a part of the speling.org system.

5.5 Conversion from current to new format

How do we convert from the current log/ds format to the new format?

- **Records without a "root" field**: Should only go to the spellings database.

[2]Due to the collection of words in "Dansk Sprognævns Retskrivningsordbog" having character of law, this is not a breach of Danish copyright law.

- **Records with both "root" and "description" fields**: The information is added to the appropriate *word* or a new *word* is created in the word database.

Alternatively we can use a simpler procedure, where a record in the log/ds format is converted to one or more records in the new format, one for each word that is already known to have that spelling, and doesn't have information which is inconsistent with the already known information about the word. If there is no word living up to this criterium, a new word is created.

5.6 Spelling-only proof-readers

What do we do about the proof-readers, who only want to consider *spellings* and not *words*? Ignore them? Feed their input through the log/ds-to-new-format converter?

6 Ideas for a new text based implementation

One benefit of using a text-based implementation can be easy caching of processing results. Another one is easy access to the data using traditional Unix tools. One serious problem with a text-based implementation is that it will require efficient directory look-ups to handle the thousands of files you will get with one (or a few) file(s) per word.

6.1 Language specification

The system will need some information about the language the dictionary covers:

- Language identification (local name, English name, ISO codes). These data can be stored in a main configuration file for the dictionary.
- A script for transcribing the language to ISO-646. The transcription doesn't have to be perfect, but it should be reasonably identifiable for people knowing both the Latin alphabet and the language covered by the dictionary. – Alternatively we should insist on using a sensible encoding of ISO-10646 on file-system level.
- List of categories the dictionary covers (i.e. "common", "physics", "cooking", "entertainment", etc.). This list should be stored in a separate file. The system should be able to generate a separate dictionary for each of these categories.

- List of word classes (i.e. “noun”, “verb”, “adverb”, etc.). This list should be stored in a separate file.
- A list of conjugation forms for each word class (i.e. “singular nominative”, “singular possessive”, “plural nominative” and “plural possessive” for nouns). These lists should be stored in one file per word class – with names derived from the word class names.
- A ranked list of authorities for each category. These lists should be stored in one file per category – with names derived from the category names.

These data should be used for validation purposes in the processing of the dictionary source, and for providing an efficient user interface for the proof-readers.

6.2 Source code

The source code (proof-reading records) for each words should be stored in a file with a name derived from its identifier in the dictionary (which again should be derived from one of its spellings and some additional information for making it unique – for example a simple sequence number).

When more proof-reading records arrive for a word, they should simply be appended to the source file for that word. The dictionary compiler can then focus its work on those words, where the source file is newer than the compiled record(s) about the word.

6.3 Dictionary compiler

(fill something in about compiling the source file for a word into a reader-friendly and a proof-reader-friendly version)

7 Ideas for a new SQL based implementation

Since SQL is generally considered to be an efficient interface for data management, one would expect a SQL based implementation of a dictionary system to be faster than a text based one.

The proof-reading data (as specified above) can only be reduced to a simple table with 5 columns, if we fold the names of the conjugation forms into the information type (for correct spellings and common misspellings).

Using a simple table with 5 columns for the source ...