FOSDEM 18

Brussels, February 2018

Anatomy of the OpenOffice localization process

And how to improve it in future

Andrea Pescetti

pescetti@apache.org

Andrea Pescetti

- Active as a volunteer in several free and open source projects in my spare time.
- These include Apache OpenOffice, where I served as Project Chair and Release Manager.
- Now mostly focusing on maintaining the (general-purpose) Italian spell-check dictionary, and on localization in general.

OpenOffice and languages

Levels of language support

Adding a language

- There are multiple meanings for what "adding a language" actually means.
- This is usually the first request we receive, and the main driver for new volunteers.
- There are document languages, interface languages and active Pootle languages.

Document languages

- Based on ISO codes of languages and countries, and MS LCID mappings (used for interoperability).
- Needs a few source code file changes. https://wiki.openoffice.org/wiki/Adding_a_new_language_or_locale
- At the end, OpenOffice will "know" that a language exists (Format ➜ Character ➜ Font, and spellcheck).

Interface languages

- Needed if you want to translate OpenOffice in your language.
- In rare cases (e.g., ca-XR or Valencian RACV, ca-XV or Valencian AVL) a language can be used as an interface language only.
- In those cases, things get complex (dictionary conflicts, hacks to fit the generic schema).

Active Pootle languages

- Out of ~120 interface languages that have a partial translation, only 41 are released, those that are 100% translated and tested.
- Those 41 and a few more have been imported into Pootle.
- The other partial translations are in the OpenOffice source code but were never imported into Pootle in the Apache era.

A new volunteer for language X

- Is X in Pootle? Easy, all done through Pootle.
- Otherwise, language X has to be created in Pootle...
- and this will be handled separately.

The tools

Overview of involved formats and tools

Pootle

- To our purpose, it is just an online PO files editor, enabling collaboration and web-based teamwork.
- It comes with command-line tools that help with management (manage.py).
- Some teams prefer to work offline with PO files (text files, POEdit or any other tool) and skip Pootle.

The ASF Pootle

- At Apache, we have one Pootle installation for all projects (not only OpenOffice).
- Committers can login directly.
- Other volunteers need to have an account created and permissions set for a specific language (L10N list).

The SDF Format

- A simple (too simple) text format to describe translations, used by the OpenOffice build system.
- Specific to OpenOffice, with many tab-separated fields.
- One interface translation is one SDF file in svn.apache.org/repos/asf/openoffice/trunk/extras/l10n/source/ (~2MB, ~73K lines/strings).

History of a string

The full process to get a string translated

1: Producing translatable content

- Source code does (should!) not contain localizable strings directly but uses resource files for that.
- Example: `main/sw/source/ui/table/convert.src`

```
String STR_CONVERT_TEXT_TABLE
    {
            Text [ en-US ] = "Convert Text to Table" ;
    };
```

- This is the English string. In other words, English is special. And this is not good.

2: Extracting localizable strings

- Generate the en-US SDF template: in `main/` run `localize -e -l en-US -f /path/to/en-US.sdf`
- All resource files are scanned.
- This should be done once per milestone. `en-US.sdf` will contain our string.

3: Generating PO template files

- `oo2po -P en-US.sdf templates`
- This uses the Translate Toolkit utilities and generates new templates to be updated.
- Templates must then be updated in Pootle (requires shell access to Pootle), paying attention to properly merging with possibly existing translations.

4: Translate!

- The new string will be available in Pootle.
- Volunteers can translate it.
- Before building, PO files must be exported from Pootle (shell access preferred, web interface available too).

5: Generate localized SDF file

- `po2oo -l de -t en-US.sdf --keeptimestamp \ --skipsource de new_de.sdf`
- The file must be placed in `extras/l18n/source/de/localize.sdf`
- The new build will use it and the new string will be translated. Any manual SDF file changes are lost in the process.

Possible improvements

Ideas for a better process

Current issues

- The last Pootle update from PO files was done a long time ago, at an unclear stage (possibly not corresponding to any releases).
- Exporting a currently 100% complete (as per Pootle statistics) translation and trying a localized build often results in errors (string validity, gsicheck). Fixing is slow and painful.
- Creating a new language from the current trunk strings can be attempted, but we would need to synchronize all languages first.

Future: Too many steps?

- The intermediate SDF format is very error prone.
- It could just be skipped and PO (or xliff) used instead.
- Other projects have already done it.

Future: More automation?

- genLang, started by Jan Iversen years ago but never integrated and currently not in active development.
- Many workflow advantages, at least in the initial design.
- An interesting approach to automating many of the manual steps involved.

Thanks!

pescetti@apache.org