

A governed translation workbench for gettext catalogues

Working notes for the LibreOffice localisation list — Karl Morten Ramberg, September 2026

Why it exists

I needed to know how good the Norwegian Bokmål translation of LibreOffice actually is, and then to fix what was wrong with it. That is not a question a translation platform answers. A platform tells you how much is translated; it does not tell you whether the translation is right, and it has no opinion at all about the 39,892 strings that are already marked done.

So this is not a tool looking for a use. It is the tool the work is being done with: every number further down came out of it while working the real catalogue, and every defect it describes was found and then fixed with it.

Two rules govern all of it.

A rule may propose; it may never decide. Every check produces a proposal that a person accepts, refuses with a reason, or amends. A refusal is recorded against the rule, not the string, because a rule that is mostly refused is a rule to fix or drop — and no per-string tool can measure that, since it never knows it asked the same question twice.

Findings are derived, not stored. A finding is a query, not a row somebody has to remember to delete. It disappears when the condition that produced it stops holding, so the list cannot drift away from the catalogue it describes.

What it can measure

Fifteen checks, each one a comparison against something that counts as evidence — the English source, or the catalogue's own majority usage — rather than a style rule somebody thought was a good idea:

PLACEHOLDER_LOST	a placeholder in the English is missing from the translation
PLACEHOLDER_ADDED	the translation carries one the English does not
PLURAL_MISSING	plural forms exist and not all are filled
UNTRANSLATED	nothing at all, in some or all of the places it is used
ACCELERATOR_LOST	the English marks a shortcut and the translation does not
SOURCE_TEXT_KEPT	a minority kept the English where their siblings translated it
CASE	the same words, capitalised two ways
MARKUP	the same words; accelerator, ellipsis or punctuation differs
TYP0	two wordings one or two characters apart
SPLIT_COMPOUND	a compound written closed here and with a space inside there
DIGRAPH	aa, oe, æ where the catalogue writes å, ø, æ
SPACE_BEFORE_HYPHEN	"PDF -visningsprogram" for "PDF-visningsprogram"
WHITESPACE_ADDED	whitespace the English does not have
BRAND_ALTERED	a product name that did not survive translation
DIVERGENT	genuinely different wordings: a decision is wanted, not a fix

The last one matters as much as the rest. `msgctxt` exists precisely so that the same English can mean two things, so a tool that treats every divergence as an error is wrong about the format. These are separated out and presented as a question rather than a defect.

Everything is reproducible from the command line, which is how the numbers in this mail were produced: the check suite over a whole catalogue, a round-trip test that re-imports

what was written out and compares, and a dictionary measurement described below.

The spellchecker, and why it is not a nuisance

A Hunspell dictionary is embedded in the editor and runs live as you type.

The reason it is usable is what it is *not* asked about. The editor does not hold a translation as a string; it holds it as a sequence of typed spans — running text, placeholders, markup, accelerator marks, escapes, invisible characters — and only the running-text spans are ever offered to the spellchecker. `%1`, `~Format`, `<emph>`, a no-break space and `\n` are never reported as misspellings, because the editor knows what they are. That is the difference between a spellchecker a translator uses and one a translator turns off within the hour.

The same span model gives the round trip for free. The spans *are* the string, so joining them back is the identity function by construction rather than by careful escaping — proven over all 39,892 strings of the real catalogue. A locked span cannot be deleted by accident: an edit that loses a placeholder is refused with the placeholder named, and an edit that changes the words rather than the whitespace is refused by comparing both sides with the whitespace stripped out.

There is a second, separate use of the dictionary that I think is the more interesting one. Instead of measuring a checker against a document with planted errors, the workbench measures **the dictionary against the catalogue**: 18,000 distinct word forms of real, human-written, overwhelmingly correct Norwegian in exactly the domain the checker is used in. A rejected word occurring once is a translator's slip; one rejected two hundred times is a hole in the dictionary. Nobody has to read the list to tell them apart, and words that also occur in the English source are reported separately — a dictionary is not wrong to reject "Bezier".

Deciding at scale

Three thousand findings is not three thousand decisions, and a tool that makes you answer them one at a time is the reason catalogue quality work does not get done.

Select any number of rows and correct them together: one wording, typed once, applied to every entry selected, with the same chip editor and the same guards around it. The dialog opens on the commonest wording among the rows rather than on whichever one you happened to click.

Decisions are recorded at one of two levels. A **string-level** decision covers every place that English string appears; a **context-level** decision covers one entry, and outranks the string-level one. So "this wording, everywhere" and "but not in this dialog" are both sayable, and the second does not require undoing the first. One decision routinely settles dozens of entries — the board shows both numbers side by side for that reason, findings and the entries behind them.

Accept and refuse work on a selection too. Each item is still checked individually underneath: a proposal computed from a wording that has since changed is refused with the reason, rather than quietly overwriting somebody's later work.

What it found

Norwegian Bokmål, the full LibreOffice catalogue — 39,892 strings. The first run raised 5,644 findings. Of those, 1,867 have since been answered and applied, and 3,012 remain open:

ACCELERATOR_LOST	2,427	WHITESPACE_ADDED	75
TYP0	249	CASE	62
MARKUP	96	SPLIT_COMPOUND	55
		SOURCE_TEXT_KEPT	29
		SPACE_BEFORE_HYPHEN	18
		DIGRAPH	1

The write-back was checked before it was trusted: the round trip over the real catalogue touched `msgstr` lines and nothing else — 1,867 changed entries across 24 files, with every comment, header and wrapping decision preserved. Re-importing the result returned exactly the accelerator count the decision log predicted.

The part the list may care about most

The largest single class of findings turned out not to be translator error.

Accelerator collisions in the Norwegian catalogue are, in the great majority of cases, **faithful reproductions of collisions in the English source**. So are a good number of the trailing spaces and the no-break spaces. Read against nothing, they look like sloppy translation. Read against the English, they are fidelity.

This cuts the other way too, and it is the useful direction: two widgets that share an accelerator in the English source are provably never visible at the same time — otherwise English itself would be broken, and it is not. The English is therefore *evidence* about what the UI actually does, and a check that compares against it can tell a real defect from a false one. A tool that reads only the target language cannot.

The upshot: a meaningful share of what any checker will flag in any language belongs upstream in the source strings, and can be reported once for every language at once rather than fixed 115 times.

Does it generalise

It was built against LibreOffice, so that question had to be answered by something else. Mageia's `manatools/dnfdragora` — 803 strings, 66 languages — was imported and the suite was run unmodified. It found 11 real defects, and it exposed one blind spot in our own code, in how a bare `&` accelerator was distinguished from an HTML entity. Nothing in the checks had to be rewritten for a second project.

What it is

A quality workbench. It answers the question a translation platform does not: is what we already shipped right, and where is it wrong.

Translator account management, approval routines and a review queue for a team are work in progress.

It has been driven hard through one catalogue in one language. That is the proof that exists today.

The stack

A JavaFX desktop client against a Quarkus server exposing GraphQL, with Kafka carrying events between services, PostgreSQL as the governed store and document storage alongside it for the material that is not relational. The checks, the catalogue importer and the writer-back are Python.

Decoupling the workbench — so that it runs against a plain Postgres database with no other dependency, and could be packaged as a LibreOffice extension — is a design goal rather than something already done.

What I would like to know

Whether the upstream English-source findings are wanted as bug reports, and in what shape.

The suite runs unmodified against any gettext catalogue, so the same measurement can be made for any language.