Persistent MT on software technical documentation - a case study

María Concepción Laguardia
Senior Information Developer
Citrix Systems
Cambridge, UK
conchita.laguardia@citrix.com

Abstract

We report on the features and current challenges of our on-going implementation of a Persistent MT workflow for Citrix Product Documentation, to increase localization coverage to 100% content in docs.citrix.com, into our Tier-1[1] languages.

By the end of 2019, we had processed seven million words of English documentation with this model, across the 24 doc sets of the Citrix portfolio (Digital Workspace, Networking, and Analytics), and raised localization coverage from 40% to 100% of the content of our documentation repositories.

The current implementation requires a process of Light Post-editing (LPE) for all languages, in order to fix over-translations, out-of-domain words, inline tags, and markdown errors in the raw output.

1 Background

The Localization team at Citrix Systems introduced Machine Translation in its documentation workflows in 2013, with an in-house implementation of Moses engines from English into German, French, and Spanish. Statistical MT was the basis for a full post-editing workflow performed by in-house linguists and a small pool of trusted contractors. With the advent of Neural Machine Translation in 2017, we switched to Google generic engines, and later on incorporated Amazon Translate and Microsoft Custom Translator. The adoption of cloud Translation Management Platforms with built-in connectors for the main generic Neural MT providers, allowed us to mature the post-editing workflows and expand them to Japanese and Simplified Chinese. This now classic workflow of MT pre-translation followed by full human post-editing has since been our default model for the documentation use-case.

Productivity increased dramatically since introducing NMT as a base for full post-editing. This had a positive impact on time-to-market and reduction of localization costs. However, the actual coverage of our localization infrastructure remained at 40% of all the Citrix Product Documentation volume, mainly into Tier-1 languages, with a few doc sets into Korean and Brazilian Portuguese. The cost of sustained localization of the remaining content using MT plus human full post-editing was prohibitively high. It was evident that just integrating MT in the workflows was not enough to produce global-ready content at the increasing speeds and volumes necessary in today's competitive market.

2 The plan

Encouraged by the better fluency and adequacy of NMT output, at the end of 2018, it was proposed to expand our localization coverage to the remaining 60% wordcount of our documentation by using raw MT (MT without post-editing).

By then we were familiar with Microsoft's successful implementation of raw MT in their technical and end-user support documentation, which Microsoft now considers ”proven for all support content types” (Schmidtke and Groves, 2019). We particularly liked some of the front-end features of

© 2020 The authors. This article is licensed under a Creative Commons 3.0 licence, no derivative works, attribution, CC-BY-ND.

[1]Citrix Tier-1 languages: German, French, Spanish, Japanese and Simplified Chinese.

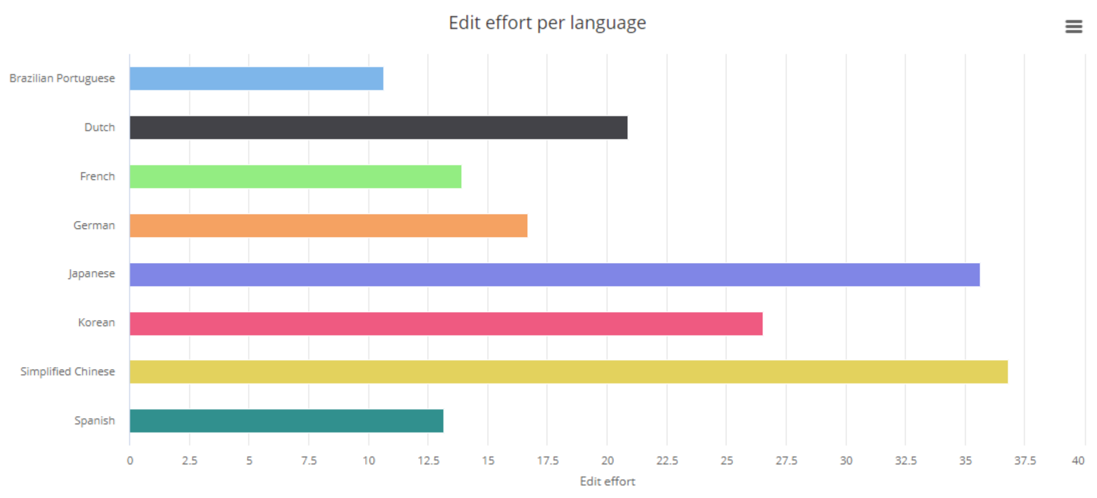

Figure 1: Edit Effort of fully post-edited documentation, measured with Okapi tools. Scale: 0-100

their deployment in docs.microsoft.com, like the switch-to-English toggle and the banner with the MT warning, and we looked at the Microsoft case as a good reference for our implementation.

We have since come across the Adobe[2] case, where similar features have been used to implement raw MT in their documentation web site.

Our focus was on **usability** as the criteria for publication, and the philosophy behind the idea was to offer this MT content as a **service** to the customer. The plan was to use the current technology we already used for full post-editing: a generic MT provider and the TMS system in place. We remained open to exploring bespoke MT engines and other TMS, depending on the results.

This solution would only be applied to technical documentation of products where the user interface is not localized, and therefore their documentation would not have been a candidate for human/fully post-edited translation in the first place.

2.1 MT *on-the-fly* vs Persistent MT

We had initially experimented with an MT *on-the-fly* implementation, which would allow customers to translate un-localized English articles on an on-demand basis, by sending an HTTPS POST request from the front-end to the Google Translate API. The main advantages of this solution were that the machine translated content would always be in sync with the latest English version, and that it would completely bypass TMS processing. The formatting of the output was also mostly correct, as the MT request was performed on the HTML output, rather than from a conversion from markdown to XLIFF, which brings its own set of problems.

However, site navigation and user experience would significantly suffer from having to send individual API requests for each topic of the English content that they would like to read in another language. Also, we would have no control over the linguistic quality of the dynamic output, and no way of fixing any quality shortcomings, should we consider this step necessary to provide usable content to our customers. There were also cost considerations: we would not be able to easily predict demand and costs of the translation API in the MT *on-the-fly* context.

In the interest of providing optimal user experience, applying a certain level of quality control, and having better visibility on costs, a decision was made to offer a solution of **Persistent MT** instead: all un-localized content in docs.citrix.com would be pre-translated into French, Spanish, German, Japanese, and Simplified Chinese using raw MT, and then published in the specific language sites of docs.citrix.com. The MT content would be permanently available to our global customers and updated regularly, alongside the human post-edited documentation sets.

3 Evaluation phase

The MT engine of choice for the Persistent MT implementation was Amazon Translate, as we had been using it successfully for the full post-edit workflow.

We found that overall quality in terms of fluency and adequacy was similar between the NMT pro-

[2]https://docs.adobe.com/content/help/es-ES/target/using/activities/auto-allocate/automated-traffic-allocation.translate.html

vided by Google and Amazon, but the latter offered a Custom Terminology feature which was crucial to solve the issue of over-translation of product names. This issue was severe in doc sets related to products where generic nouns are used as part of the product name (for example: ”App Layering”, ”Workspace”, ”Endpoint Management”).

In order to collect data on the current quality gap between the raw output of generic NMT and our human translations, we started measuring Edit Effort[3] in the full post-edit workflow. The data thus obtained corroborated our human assessment of the quality that could be expected from the Amazon generic MT engine on the Tier-1 languages (see Figure 1). Edit Effort was chosen over BLEU scores, as we understand BLEU tends to underestimate the quality of NMT (Shterionov et al., 2017).

We also ran quality assessment checks on the MT output using a popular QA tool, Xbench[4], in order to scrutinize the raw MT output and identify quality shortcomings, error patterns, their estimated frequency, and their severity.

In addition, we ran a human evaluation test on 55 articles (topics) of Citrix documentation, where in-house linguists, contractors, and native speakers of the target languages were asked to rate the quality of the MT output with a criteria of *usability*, using scores from 1 to 5.

The scores were to be given at topic-level, rather than segment-level. The idea was to assess whether the translated content would be enough for the user to complete the task described in a particular topic. Evaluators were asked to penalise output that would confuse the user to the point of being unable to follow or understand the text. The averaged result across all 55 topics was 4 for the European languages, and 2 for the Asian languages. However, there were some discrepancies in the ratings, consistent with the subjectivity of human evaluation based on *usability* criteria.

4 The implementation

The Persistent MT workflow uses the current TMS already in place for the human/fully post-edited documentation workflow. This simplifies both the MT processing and the leverage of human quality translations from our Translation Memories.

[3]Edit Effort is a measurement based on edit distance and fuzzy match scores of two samples. `https://okapiframework.org/wiki/index.php/Translation_Comparison_Step`

[4]`https://www.xbench.net/`

The Persistent MT workflow is very similar to the default full post-edit workflows: source files are taken from Bitbucket repositories in markdown format and uploaded to the TMS, where the Translation Memory is leveraged before processing new content with MT. The target files are then downloaded from the TMS and uploaded to Bitbucket for publication in docs.citrix.com. Unlike the default human post-editing workflow, only in-context and 100% matches are leveraged from the TM. All fuzzy matches (99% and lower) are sent to MT for processing, together with new words.

In order to maximize the usability of the machine-translated content, and to manage customer expectations, the documentation website features the following:

- **Explicit warning and legal disclaimer**, to inform the user that the content is MT output.

- **Switch to English** toggle, to direct the user to the English version.

- **Parallel browsing**: a pop-up window that allows the readers to see the underlying English source of a paragraph when they hover over the text.

- A *verbatim* **customer feedback** form, where we ask the user whether the translated content was useful or not, and allows them to enter comments when the answer is ”No”. We are aware that in other implementations like Adobe’s or Microsoft’s, there is no option to enter further feedback about the translation, only a ”yes/no” reply. However, we have opted for allowing customers to enter comments, so that we can get some insights on their perception of quality. It may also guide any action on our part to improve the service (for example, a targeted PE/fix process based on specific feedback).

These MT docs usage data is collected via Google Analytics *events*. Since the addition of the customer feedback feature is fairly recent, we are unable to report significant insights at this time.

5 The challenges

5.1 Regular updates

Sustaining the increasing volumes of localized documentation updates is the main challenge we face at the moment, as the default process of manual file handling and TMS job creation is not scalable to the Persistent MT workflow. For this purpose, we are currently developing an automation platform that will smart-trigger localization

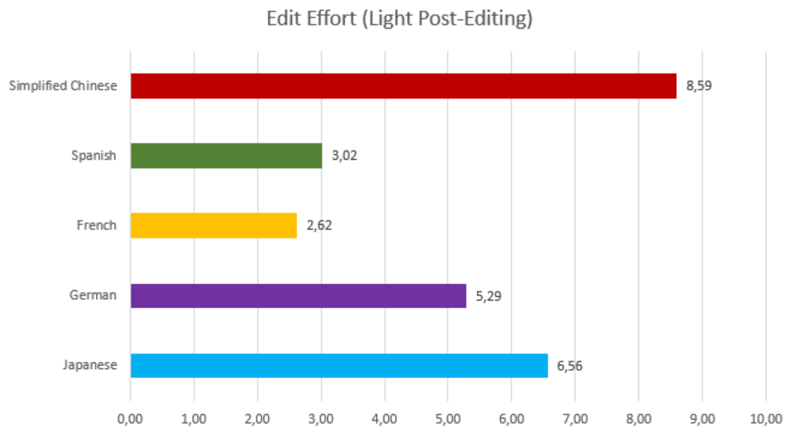

Figure 2: Light Post-Editing effort measured with Okapi tools. Scale: 0-100

jobs from the Bitbucket source repositories, based on the volume of changes in the English content and/or a predefined update schedule.

The platform is meant to be TMS- and MT-engine agnostic. This will facilitate the use of any MT engine available (not only the ones currently available in the TMS). Also, it will allow us to overcome TMS bottlenecks that may happen when increasing volumes of documentation are sent for localization simultaneously.

5.2 Output

We are also facing some challenges with regards to the actual quality obtained from generic MT engines when processed through the TMS and rendered as HTML in docs.citrix.com.

Processing the MT via third-party platforms and having no control over the quality delivered by third-party engines introduces a lot of uncertainty in our process, and it forces us to implement continuous quality estimation on each output. There is a difficulty in finding the root cause of some issues on the output, as it is unclear whether they come from the actual MT engine, or whether they are due to specific TMS-processing choices (markdown to XLIFF parsing, the way the MT API is called, and so on):

1. Inline tags: normally correspond to markdown formatting for highlighted text and URLs (for example, double-asterisk for bold text). These can be misplaced in the MT output, and we currently have no means to automatically fix the target. We have observed a disparity of tag placement behaviour when processing MT via different TMS systems, which seems to point at the root cause for these issues being on the TMS processing side, rather than the MT engine behaviour.
2. Whitespace added or removed: this can completely break the markdown formatting. For example, when spaces are added after and before the double-asterisk for bold text.
3. Terminology deviations and out of domain words: some non-translatables and brand names are currently successfully handled with the Amazon Custom Terminology setting. We also keep blacklists containing some predictable out-of-domain words that arise when polysemous words appear in the source. However, the uncertainty of the output remains, as we cannot possibly blacklist every out-of-domain word that generic MT can produce. Even when we can detect terminology deviations, we are not capable of automating fixes for them in all cases as this would require a more complex solution than a simple search and replace mechanism.
4. Total or partial over-translation of file paths, command names, and parameter names. We are working on implementing a pre-processing step in order to mask file/command line paths in the source files, where possible.
5. Other issues: spurious characters added in Asian languages, and rarely, wrong target language for a segment (Portuguese instead of Spanish, for example) or non-existent (*made-up*) words in the target.

Most LPE and post-processing efforts (see figure 2) go into fixing issues 1 and 2 above. While the volume of LPE and post-processing work is small, the importance of these manual edits is considered high as they may impact usability (over-translated parameter names, file paths) and the look and feel of the content.

5.3 Quality definition

The very definition of what is *publishable* quality has also been challenging. Without any data from customer feedback to establish a baseline on what is the expected quality for machine-translated documentation, we relied on the in-house Citrix Translation Services team to develop the criteria and associated thresholds that define quality for the Citrix Persistent MT use-case. These criteria were then implemented in the form of specific linguistic checks used for quality estimation (see subsection 5.4)

The adequacy of these quality criteria still needs to be confirmed by actual data from the customer feedback mechanism in place.

5.4 Quality estimation

Quality is estimated based on linguistic features that can be detected using Checkmate, the quality assessment tool available in the Okapi Framework.

We apply language-specific custom checks (non-translatables lists, blacklists for banned or non-preferred terms, alphanumeric mismatches, known error regex patterns, camel-case word mismatches, and so on).

We have expanded the default Checkmate code in order to integrate these checks into the automated delivery platform of Persistent MT under development.

In order to deal with terminology deviations and non-existent words appearing occasionally in the output, we are considering a process which will compare each MT output to a normalized reference vocabulary obtained from the human-translated documentation, and flag any differences between them.

We are also implementing a scoring system, in order to produce a quality report for each documentation update, which will include the error categories found, and the associated quality score, per category and for the whole doc update. The score for a doc set will be based on an overall *translation issue density* value: weighted issues per 1000 words. The higher this value, the lower the quality score.

The idea is to eventually be able to use this report as a quality gate for publication. This would allow us to directly publish the content if the score is optimal, or else re-route to a process of targeted PE if the score is too low.

However, we are aware of the limitations of this method of estimation, as it relies on predefined issues (leaving out potential problems in the output that we cannot foresee). Also, human intervention is necessary to review these reports and discard false positives. It also relies on human-assigned, subjective, scores, which can be difficult to fine-tune.

This naive implementation of linguistic checks in order to estimate output quality is a short-term solution, as we expect to rely on machine learning-based QE in the mid-term.

Our challenges in this area are the lack of QE tools readily available to plug into our workflows. As such, we are researching some available open-source frameworks (Quest++, OpenKiwi).

However, these ML-based tools require some ML and engineering expertise in order to use them reliably. They also require substantial efforts in building the right data pipeline and infrastructure to sustain the necessary workflows for model training/updating that will help us implement ML-based QE with confidence.

Ideally, we would wish to have a sentence and document-level QE tool that can be easily used by our in-house Translation team, to train and evaluate models by uploading our Translation Memories, without requiring substantial training in ML or coding experience.

6 Conclusion

We have achieved localization coverage of 100% of Citrix product documentation using Persistent MT for Spanish, German, French, Japanese, and Simplified Chinese. However, sustaining the update cadence of the English documentation proves challenging without an automation platform, and without a solid quality estimation process in place for each MT output. The same factors make scalability into other languages difficult.

Besides, we are not truly ready to move to a raw MT without review workflow. A process of LPE is still considered necessary, even for the languages where linguistic quality, fluency, and adequacy of the raw MT is very good. This LPE process targets mainly inline tags, over-translations, and blacklisted terms. We are aware that other successful implementations of raw MT in docs, like

Microsoft's, rely on a tighter control and customisation capabilities of their MT engines, and this is a crucial difference between their case and ours. In this respect, we are currently testing the customisation capabilities of Google AutoML and Microsoft Custom Translator, to train English-Japanese and English-Chinese engines.

In addition, due to the constraints of the source format (markdown), we have encountered further limitations in the quality of the raw output. The markdown errors in the output affect formatting and rendering, and some of them cannot be fixed with automated processes. They require human intervention to pass the validators in place, before publication in the docs web site.

Quality estimation remains an important challenge, as we observe a lack of readily available tools that we can integrate into MT workflows in a production environment. As a workaround, we use current QA tools in the market to perform *estimation* tasks, but these tools are best used in traditional localization workflows where the quality checks are aimed at helping human intervention (PE), rather than bypassing it.

A TMS-agnostic, reliable and customisable QE tool that we could plug in our current workflows would significantly reduce the burden on our Translation Project Managers and post-editors, and accelerate the end-to-end workflow by allowing the implementation of an automated quality gate.

Also, we still depend on a third-party TMS to process the MT content. This dependency adds a certain degree of uncertainty to our processes and restricts the MT engines we can test and use for optimal output.

We believe an in-house custom tool to support the entire Persistent MT pipeline would be beneficial and this is part of the automation platform we are currently working on.

References

Amazon Translate. 2019. Amazon Translate Developer Guide. *https://docs.aws.amazon.com/.*

Felice, Mariano. 2012. Linguistic Indicators for Quality Estimation of Machine Translations. *Master's thesis. University of Wolverhampton, UK.*

Google Translation. 2019. Cloud Translation Documentation. *https://cloud.google.com/translate/docs.*

Kepler, Fabio, Jonay Trénous, Marcos Treviso, Miguel Vera, and André Martins. 2019. OpenKiwi: An Open Source Framework for Quality Estimation. *https://arxiv.org/abs/1902.08646*

Microsoft Custom Translator. 2019. Custom Translator documentation *https://docs.microsoft.com/en-us/azure/cognitive-services/translator/custom-translator/overview*

Okapi Framework. 2019. Okapi Documentation Wiki *https://okapiframework.org*

Schmidtke, D. and Declan Groves. 2019. Automatic Translation for Software with Safe Velocity. *Electronic proceedings of MT Summit, Dublin 2019.*

Shterionov, Dimitar, Pat Nagle, Laura Casanellas, Riccardo Superbo and Tony O'Dowd. 2017. Empirical evaluation of NMT and PBSMT quality for large-scale translation production. *Electronic proceedings of EAMT 2017 (User track).*

Specia, Lucia, Gustavo H. Paetzold, and Carollina Scarton. Multi-level Translation Quality Prediction with QUEST++. 2015. *Proceedings of ACL-IJCNLP 2015 System Demonstrations*. Association for Computational Linguistics and The Asian Federation of Natural Language Processing.

Specia, Lucia, Carolina Scarton, and Gustavo H. Paetzold. 2018b. Quality Estimation for Machine Translation. *Synthesis Lectures on Human Language Technologies*. Morgan & Claypool Publishers.

Way, Andy. 2013. Traditional and Emerging Use-Cases for Machine Translation. In: *Proceedings of Translating and the Computer 34, London*