

Open source software in translator training

Marcos Cánovas
Richard Samson
Universitat de Vic



ABSTRACT

Translator training implies the use of procedures and tools that allow students to become familiar with professional contexts. Specialised open source software includes professional quality tools and procedures accessible to academic institutions and distance students working at home. Authentic projects using open source software and crowdsourcing procedures are indispensable resources in translator training.

Keywords: translator training, open source software, computer-assisted translation, collaborative learning, socioconstructivist approach, translation projects, crowdsourcing

RESUM (*El programari lliure en la formació del traductor*)

La formació de traductors implica l'ús de procediments i eines que permetin els estudiants familiaritzar-se amb contextos professionals. El software lliure especialitzat inclou eines de qualitat professional i procediments accessibles per a les institucions acadèmiques i els estudiants a distància que treballen a casa seva. Els projectes reals que utilitzen software lliure i traducció col·laborativa (crowdsourcing) constitueixen recursos indispensables en la formació de traductors.

Paraules clau: formació de traductors, software lliure, traducció assistida per ordinador, aprenentatge col·laboratiu, enfocament socioconstructivista, projectes de traducció, traducció col·laborativa

RESUMEN (*El software libre en la traducción del traductor*)

La formación de traductores implica el uso de procedimientos y herramientas que permitan a los estudiantes familiarizarse con contextos profesionales. El software libre especializado incluye herramientas de calidad profesional y procedimientos accesibles para las instituciones académicas y los estudiantes a distancia que trabajan en su casa. Los proyectos reales que utilizan software libre y traducción colaborativa (crowdsourcing) son recursos indispensables en la formación de traductores.

Palabras clave: formación de traductores, software libre, traducción asistida por ordenador, aprendizaje colaborativo, enfoque socioconstructivista, proyectos de traducción, traducción colaborativa.



1. Introduction: The training context

1.1 Professional translator profiles and skills

Translators nowadays may have to cover a wide range of functions within complex team workflows:

- Translator. The most traditional, and still central, role of a specialist in translating texts between certain languages.
- Localiser. Usually taken to indicate a translator specialising in on-screen and associated translation work in a wide variety of formats, and skilled at language engineering tasks such as format conversion and filtering. (In this document, however, we have used the terms translation and localisation interchangeably.)
- Proofreader and editor. A specialist in the preparation and supervision of texts.
- Project manager. The person in charge of managing a team translation project, skilled at communication and briefing team members.
- Terminologist and resource manager. A specialist in preparation and maintenance of glossaries, translation memories and style guides.

Translation trainees need to have pre-service opportunities to experience these different roles in team projects.

1.2 Tendencies in translating procedures: crowdsourcing

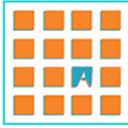
Many real world projects seek volunteers to help with localisation, nearly always from US English into other languages. In this crowdsourcing approach to translation work, sometimes referred to as community translation, the translators are volunteers, not necessarily specialist translators. The philosophy behind crowdsourcing, which was used in other fields before translation, is that by making an open call to undefined volunteers those best able to contribute to the solution of the problem at hand will self-select themselves. The great majority of crowdsourcing localisation projects are themselves open source, since it is precisely these projects that have the open architecture that facilitates collaboration. There have, however, been cases of proprietary projects that have resorted to users to localise their application through Web 2.0 networking, the most well-known case being Facebook.

Space does not permit us to explore the different approaches to crowdsourcing localisation and workflow. The interested reader may refer to *Revista Tradumàtica 08. Localització i web* (2010) and the *Open Translation Tools* manual (2011).

We do, however, wish to remark on the importance of this kind of localisation and how it can be harnessed in translation training, because the initial reaction of many professional translators to crowdsourcing is rather negative. For them, the approach is seen as a threat that undermines the profession by putting unpaid translation work in the hands of untrained volunteers.

The advantages of crowdsourcing for the project concerned include:

- users know best (and therefore make optimal translations)
- the costs of translation are low
- translation work and integration is fast (turnaround time)
- the process enhances user identification and involvement with the project
- peer review and editing of translation is built into the system



Where volunteers may be weak in some skills, they can be supported by online documentation of the translation process, contact with others working on the same project, sandbox environments for experiments, style guides, etc. In fact, the chance to work in a well-articulated crowdsourcing localisation project may be the most sophisticated kind of project that translators in training can hope to experience, and is fully coherent with state-of-the-art professional approaches to team translation projects. Consequently, participation in this kind of project has far-reaching benefits for students in terms of professional preparation because the approach so closely mirrors current models of professional work. Furthermore, students can gain public acknowledgement for their work in these projects, which will be in day-to-day use in the real world. Reducing or eliminating the distance between the ivory tower world of academia and hands-on language in use is an important factor in motivation for students.

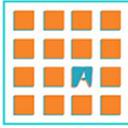
Far from undermining the profession, crowdsourcing of localisation projects helps participants to collaborate more effectively in a controlled way that is, among other things, a useful point of access to professional translation work. It is not the case that crowdsourced translations are reducing demand in the market, since these projects would never be offered commercially anyway. Professional translators who feel threatened by crowdsourcing should recognise that its use is generally limited to on-screen translation of community resources. Meanwhile, the professional translation market worldwide continues to grow at a spectacular rate, according to analysts such as Common Sense Advisory.

1.3 Computers in translation training

As in many other fields, the advent of powerful pervasive computing has revolutionised professional translation, which is now unthinkable without computers. Computers are used not only as writing tools (word processors), but for communication (e-mail, forums, etc.), documentation and searches (dictionaries and glossaries, web pages, blogs, etc.), data storage (local hard drives, cloud computing, etc.) and there are many specialised digital tools for translation (glossaries, computer-assisted translation, aligners, format converters, corpus tools, etc.).

For students to become familiar with current translation practice, computers must be fully integrated into translation training programmes. The centres that organise such activities must have a clear idea about the nature of the challenge. Aspects to be taken into account include the following:

- Approach. The use of computers will depend on the pedagogical approach, which means that an initial discussion on teaching methods and techniques will be necessary.
- Classroom facilities. Apart from the distribution of furniture and space, allowance must be made for hardware diversity, with students using desktop computers belonging to the institution or personal laptops, for instance. If students are to use laptops, there must be sufficient mains electricity sockets and wireless internet access.
- Distance learning or blended learning. In these cases, students will most likely work exclusively with their own computers.
- Operating systems. Microsoft Windows is the most common operating system in personal computing, but on centre computers, at least, it may be worth installing Linux too, and some students may have Mac computers. Such diversity in operating systems is a challenge for teachers because the computing tools available are different in each case.
- Computer programs. Centres must choose between open source and proprietary software (see next section) and the programs that are most suitable for the training objectives. Centres must also choose which version of programs to use. These are not isolated decisions. Questions of operating systems and whether students will use their own computers (either in the classroom or at home) must be taken into consideration here.



Many of the issues mentioned above have economic consequences. For this reason, among others, training centres should publish and maintain a computer policy document outlining privileges and responsibilities for computer use, so that students and teachers know where they stand from the outset.

2. Learning environments and open source software

2.1 Why open source software?

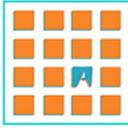
Educational institutions must design training programmes that allow students to work in quasi-professional translation environments. It may be desirable for students to undertake authentic projects (see Kiraly: 2002), but this is not always possible. Whatever the case, class projects should call on student-translator skills and technical resources that are similar to those used in authentic professional projects.

Computer resources in professional projects typically include proprietary applications, such as those of computer-assisted translation (CAT), like SDL Trados, Star Transit, Déjà Vu and Wordfast. Commercial licences for these programs cost three-figure sums. Academic institutions that wish to use them will need to contact the program vendors to obtain educational licences for the centre and for students (including distance students). Such licences are, in general, more easily available than previously but policies change from year to year and involve a degree of vulnerability for centres, which naturally wish to ensure continuity in their training programmes.

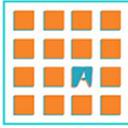
In contrast, open source software offers an accessible training (and professional) alternative. Each program will have its own particular design and characteristics, but basic program families exist in proprietary and open source versions, with broadly similar characteristics. For instance, both Microsoft Word (proprietary) and OpenOffice Writer (open source) are word processors: their interfaces differ, but both allow students to understand what word processors are and what they can do. Knowing either makes it much easier to understand the other. Similarly, open source CAT programs, like OmegaT or Anaphraseus, offer the same basic tools as proprietary alternatives, but they are free.

There are many examples of open source alternatives to commercial translation programs. Generally speaking, they are less sophisticated but they may be extremely helpful in training contexts. (Indeed, this distinction is important across all educational fields, not just in translation. Sophisticated and expensive commercial applications may be state-of-the-art but at the same time offer far more functionality than is required in a training context.)

In short, the advantages and disadvantages of each kind of software used for translator training may be summarised as follows:



Advantages	
<p>Open source software:</p> <ul style="list-style-type: none">• Free. Accessible to institutions and students at home.• Suitable for learning and professional environments.• Solutions equivalent to those of proprietary software.• Use of open standard formats (like TMX for translation memories).• Unrestricted for modification or distribution, which means rapid evolution, intensive testing and a sustainable project.• Often available in equivalent versions for different platforms (Linux, Microsoft Windows, Mac)• The Linux operating system can be installed alongside Microsoft Windows or Mac OS, at no cost.	<p>Proprietary software:</p> <ul style="list-style-type: none">• Educational licenses (free or low cost).• Suitable for learning and professional environments.• Extensively used in professional projects.• Powerful tools to control complex projects with multiple languages and several translators.• Import and export from standard formats.
Disadvantages	
<p>Open source software:</p> <ul style="list-style-type: none">• Less used in authentic professional projects.• Less powerful than some proprietary programs when dealing with complex projects that include multiple languages and several translators.• Depending on the project, poor versioning of the program and localised versions.• Depending on the project, poor documentation.• Depending on the project, insufficient functionality.• Depending on the project, discontinued development.	<p>Proprietary software:</p> <ul style="list-style-type: none">• Expense. The cost of the licence includes advanced functionality that is rarely used in classroom projects.• Restricted distribution with licensing per computer.• Proprietary formats (although it is generally possible to export to and import from standard formats).• Many applications depend on other proprietary software (such as Microsoft Windows and Word).



The open source concept includes a preference for collaborative knowledge and development, open to all, as opposed to knowledge that is restricted and exploited for profit. Choosing one model or another has these implications too, and academic institutions have an ethical responsibility here to promote open source solutions in order to remove barriers to learning and facilitate access to knowledge. Bearing these considerations in mind, and on the basis of our experience in class, we believe that open source software should be the preferred, fully valid, professional option for pre-service translator training, both in classroom and distance learning courses.

2.2 Integration of open source software in learning environments

There is in principle no appreciable downside to using open source software, provided that the programs involved are well chosen. Students may already be using prestigious open source programs such as Firefox, Audacity and OpenOffice without even being aware of their provenance.

In the case of less well consolidated open source projects, teachers should be aware of the need to provide a high degree of local support to students, to compensate for any shortfall in quality of project documentation and versioning. Teachers need also to make sure that selected programs are available in versions for the operating systems in use, and if necessary that guidance on installation of an open source operating system is provided. As mentioned previously, the training centre should ensure that hardware and software requirements are explicitly stated in a policy document made available to students before course enrolment.

The programs and activities that we propose below allow the integration of the professional procedures and pedagogical approaches described in previous pages. With this collaborative approach to teaching and learning, students can experience several roles (translator, editor, project manager, etc.) in authentic projects. Open source applications, a collaborative approach to learning and crowdsourcing processes are complementary strands to a more social and less linear approach to education and professional activity that is in tune with the rapidly changing problem-solving nature of our times.

3. Programs and translation projects

3.1 Programs

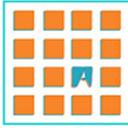
There is an enormous and constantly expanding range of open source software to choose from to undertake every imaginable task: text editors, word processors (OpenOffice), computer-assisted translation (OmegaT, Anaphraseus, bitext2tmx), automatic translation (Opentrad, Apertium), audiovisual translation (Subtitle Edit), gettext translation projects and tools, crowd sourcing tools, etc. Specific projects may develop their own project tools, forums, web interfaces and workflows for localisation.

The full range of tools available is reflected in other sections of this publication.

3.2 Suitable objects for localisation training projects

Authentic translation projects for proprietary software or confidential documents (of the commercial translation market) are not generally available for exploitation in student translation projects. Open source projects offer not only powerful tools for translation but are also themselves, in their interfaces and documentation, suitable objects for student translation projects. There are a great many text-based file formats available for localisation:

- XML language files. Many desktop programs use XML files to store interface language items. An example is Notepad++, which we comment on further in the next section.



- PHP scripts, PO files, gettext. For example, Wordpress themes and plugins may be localised with this method.
- XUL Firefox browser extensions.
- Wiki project documentation. Many open source projects maintain wiki knowledge bases that are available for localisation, such as Sugar Labs at <http://wiki.sugarlabs.org>
- Wikipedia. The Wikipedia in different languages are independent but in practice most Wikipedia articles in other languages are inspired by the corresponding English language article.
- Open source project manuals. For example, FLOSS Manuals at <http://translate.flossmanuals.net>
- Film subtitles. There are many web resources for reading and writing subtitles, such as at <http://www.opensubtitles.org>
- Online project translation portals. There are many projects that syndicate their localisation efforts through online portals, such as Launchpad at <https://translations.launchpad.net>

3.3 Design of open source translation projects

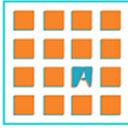
As we have mentioned, the teaching approach will determine how computer tools may be used in classroom activities. The translation projects that we propose here are based on a socioconstructivist approach, where students are at the centre of the learning process; the teacher is not considered a knowledge source controlling everything, but a *facilitator* that provides the students with *scaffolding* to help the learning community negotiate the cognitive changes that constitute the learning process. Moreover, the students do not work alone: the social factor, communication among members of the group, means that each individual learning process is supported by contributions from other participants (Vygotsky, 1978; Lantolf, 2000).

Collaborative translation projects are well suited to a socioconstructivist approach (Kiraly, 2000): the teacher proposes projects and gives the students the information and resources they need as a support in their learning process. The students make teams and undertake their project, dividing up the work according to the professional profiles we have mentioned previously: project manager, terminologist, translator, editor, proofreader. Ideally, the students should work on authentic projects but activities designed for the classroom, simulating professional tasks can also give excellent results.

To give some examples of activities following this approach with open source software, we will propose some specific tasks that may be included as parts of projects and give examples of complete translation projects that students can undertake.

3.3.1 Tasks using open source software

- Machine translation as part of the workflow. Through machine translation students get a draft version (pre-translation) of the target text that they can edit to produce the final translation. Opentrad, Apertium or Moses are available resources.
- Manipulating translation memories. Translation memories are usually the result of a translation task, but can also be prepared in advance to pose certain challenges to the students. A translation memory can be provided with solutions for only certain parts of the text to be translated, and these excerpts may not have a 100% match with the text being translated. The teacher can prepare segments with translated sentences which are slightly different from the sentences that the students must translate, and they will have to detect



differences and solve problems. Open source CAT tools that can be used include OmegaT and Anaphraseus. Translation memories may be edited with Olifant (included in the Okapi Framework platform, developed in Java, but Olifant has not been developed in Java yet and it is at present only available in the old .NET implementation, i.e. for Microsoft Windows only). Translation memories in TMX can be edited with any plain text editor. Finally, bitext2tmx is a text aligner that can generate translation memories adapted to teaching purposes. This application creates memories in TMX from a .txt file containing the source text and another file with the target text.

- Merging translation memories. The source text is divided in several parts. Each student or group translates a part with a CAT tool and generates a translation memory. Each group then translates the whole text using with the other translation memories (which can be merged into one memory with Olifant), accepting or changing the pretranslated sentences offered by the tool, to edit the final translation. The whole class can also discuss this final version.

- Glossaries. CAT tools include glossaries for the translator. Identifying terminology and proposing a suitable translation is a key task, especially with respect to certain typologies of texts (i.e. technical or scientific documents). By generating and using glossaries, students learn how to deal with a powerful tool that makes their task simpler and more consistent (the translator accedes easily to equivalences, with the certainty that the translation for each technical concept is always the same). Tasks may include working with a glossary proposed by the teacher or client, which students should respect in their translations.

There are various other technical aspects that teachers should bear in mind when considering which translation projects to propose to students, since they affect the degree of difficulty of the work.

- Character encoding. Students and teachers need to be aware of the basics of character encoding, in order to avoid unexpected problems, particularly with characters not included in the unaccented Roman alphabet.

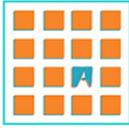
- Localisation of desktop or server-based applications. Localisation of open source desktop applications means that students can see their translations in action on their own computers autonomously. Server-based applications such as wikis also allow students to see the results of their work immediately. Other applications with server-client architecture, such as Wordpress, may require installation on a local server for translation work to be tested online.

- Computer operations. There is a great degree of difference in complexity between the various localisation projects and workflows. Most student translators will prefer to work in contexts that challenge their translation skills without over-stretching their computing skills: XML, Launchpad, subtitling and wiki projects. Teachers need to be aware of this because to a large extent it is likely to be up to them to provide computing support to students. If in doubt, in training texts, it is best to avoid projects that require numerous complex non-linguistic operations.

- Field of translation. As we have seen, many authentic open source translation projects are in the field of computing itself (interfaces and manuals). Most students working in this field will need considerable preparation to enhance their understanding of computer applications and suitable terminology.

3.3.2 Example translation projects

- XML localisation. Brief instructions: Translate the English XML language file of Notepad++ into another language using a text editor. Consult the Notepad++ translation web page for further guidelines. Ensure that your use of the the Windows menu keyboard shortcut



code "&" is appropriately used. Work in pairs as translator and proofreader. You may choose to do half the translation each in each role. Write a detailed commentary on your work, with rationales for your decisions. If a published translation is available, compare and contrast it with your own work. Test your finished XML file in production, ensuring that there are no bugs and that space restrictions are respected. Publish your work and commentary in the online classroom.

- Subtitle translation. Brief instructions: Find a film and external subtitles for it on OpenSubtitles.org or a similar website. Check that you can play the film and subtitles in the VLC media player, and that the subtitles correspond to the film precisely. Work in pairs as translator and proofreader. Choose 10 minutes of the film for each of you to translate. Translate the corresponding parts of the external file (in a text editor or in a subtitle editor of your choice). Send your work to your proofreader. Write a detailed commentary on your work. Publish your work and commentary in the online classroom.

More ambitious translation tasks to work on with students involve participation in online open source projects. Space does not permit us here to go into detail, but the outlines below for translating Firefox browser add-ons, Wordpress blog plugins and the Drupal CMS core and Drupal projects should give an idea of the range of possibilities.

Teachers will need to adapt the suggested localisation workflows in some cases, in order to give students a rather more shielded educational environment. Teachers will also need to be able to install server applications to work on Wordpress and Drupal.

- Firefox add-ons:

Register at www.babelzilla.org

Download add-on translation files for projects of interest.

Distribute them to students for translation offline.

Test the translations in Firefox locally by changing the Firefox locale.

Optionally, register at Babelzilla as translator for the projects concerned and upload the translations.

- Wordpress plugins:

Set up a Wordpress test site.

Install any plugins of interest.

Use the "Codestyling Localisation" plugin to generate the corresponding .po files.

Distribute the .po files to students for translation with PoEdit (or another offline tool).

Receive the translations and load them up to see how they look.

Optionally, contribute them to the plugin project.

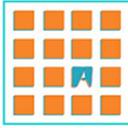
- Localising Drupal:

Install the localised Drupal distribution (http://drupal.org/project/l10n_install).

Install some Drupal projects (<http://localize.drupal.org/translate/projects>).

Translate Drupal and its projects.

Optionally, contribute the translations back to the community.



4. Future prospects

In pre-service translation training, open source tools in general offer the best opportunity for authentic accessible project work in diverse situations. The main features of open source software (open standards, collaboration, online communities, shared resources, sophisticated workflows and administration) are increasingly important in commercial translation contexts that use proprietary tools, subject to various restrictions. This synergy is likely to increase in the future. Already we observe that:

- Working on translation projects in distributed teams across the Internet is standard practice.
- Sophisticated workflows and resource management for collaborative teams tend to require the use of open standards. Open source projects have experience in collaborative architecture for programming. Extending these frameworks to facilitate and manage translation harnesses this expertise.
- Shared online corpus language data is becoming more important than local translation memories and isolated knowledge databases.

Worldwide the demand for translation continues to grow year on year and the sector needs to respond to this situation with ongoing improvements in quality, turnaround time and inclusion of minority languages in order to make more information available to users in more languages. Translators will need to become ever more proficient in the supervision and control of integrated systems, including terminology management, pre-editing and pre-translation, post-editing, proofreading and collaborative work in distributed multidisciplinary teams. Pre-service training in open source translation projects is in general the most coherent approach and best opportunity to meet this challenge.

5. References

5.1 Bibliography

Open Translation Tools (2011). Available for download at <http://en.flossmanuals.net>

Cánovas, Marcos; Samson, Richard (2008): "Dos ejemplos de aplicación del software libre en la docencia de la traducción", in Diaz Fouces, Oscar; García González, Marta (eds.): *Traducir (con) software libre*. Granada: Comares, p. 193-210.

Kiraly, D. (2000). *A Social Constructivism Approach to Translator Education*. Manchester y Northampton: St. Jerome Publishing.

Lantolf, J. (ed). (2000). *Sociocultural Theory and Second Language Learning*. Oxford: Oxford University Press.

Revista Tradumàtica 08. Localització i web (2010). Available at <http://www.fti.uab.es/tradumatica/revista/num8/sumari.htm>

Vygotsky, L. S. (1978). *Mind in Society*. Cambridge, MA: Harvard University Press.

5.2 Web sites

- Apertium: <http://www.apertium.org/?lang=en>
- bitext2tmx: http://bitext2tmx.sourceforge.net/index_en.html
- Common Sense Advisory: <http://www.commonsenseadvisory.com/Resources/FactsandFigures.aspx>
- Drupal: <http://drupal.org/>



-
- Firefox add-ons: www.babelzilla.org
 - FLOSS Manuals: <http://translate.flossmanuals.net>
 - Launchpad: <https://translations.launchpad.net>
 - Olifant: <http://sourceforge.net/projects/okapi/files/>
 - Okapi: Framework <http://okapi.opentag.com/>
 - Open Subtitles: <http://www.opensubtitles.org>
 - Opentrad: <http://www.opentrad.com/en/inicio>
 - Moses: <http://www.statmt.org/moses/?n=Public.Demos>
 - Sugar Labs: <http://wiki.sugarlabs.org>
 - Wordpress: <http://wordpress.org/>