Localisation Standards for Joomla!
Translator-Oriented Localisation of CMS-Based Websites

Jesús Torres del Rey[1], Emilio Rodríguez V. de Aldana[2]
[1]Department of Translation and Interpreting
[2]Department of Computer Science and Automatics
University of Salamanca
Spain
jtorres@usal.es, aldana@usal.es

Abstract

For a localiser, the shift from static to CMS-based dynamic websites usually involves assimilating a new editing environment, acquiring administrative rights for the site, and relinquishing the various benefits of using CAT tools. However, the possibility of integrating CAT tools in the localisation process is now becoming a reality by means of localisation standards (mainly ITS and XLIFF). In this paper, we introduce an experimental Java application we have developed for the import/export of multilingual web content for the Joomla! CMS (with the FaLang extension). We go through the workflow and explain the lessons learnt from our experiments with this and other related tools. As our research is translator-oriented, we discuss some current limitations for localisers' work in the theoretical and practical approaches taken for the multilingual management and translation of CMS-based websites and suggest some alternatives for the future.

Keywords: *web localisation, localization, Content Management System, CMS, standards, Internationalization Tag Set, ITS, XLIFF, roundtrip, interchange, Translation-Oriented Localisation Studies, communication, text, meaning*

1. Introduction

The development of websites has quickly evolved over the last half decade, as Esselink (2002, p.5) announced for digital content in general, from being "traditionally written" using html editors, such as Dreamweaver, FrontPage, Expression Web, Amaya, Nvu or Kompozer, to being "dynamically built using database driven publishing systems or content management systems", particularly thanks to the boom of FOSS web CMSs such as Drupal, Joomla! or Wordpress (Torres del Rey and Rodríguez V. de Aldana 2011, 2014).

On the client side, it could be argued that things have remained essentially the same for the last two decades due to the consolidation of HTML as the main content language and file type, and of browsers as the leading web surfing application. Of course, user *experience has* changed dramatically with the introduction of more dynamic client-side technologies such as Javascript and other scripting languages, Ajax or CSS, the embedding of Java applets or flash animations (Mata Pastor 2005, pp.197-198], the gradual move to XML vocabularies and HTML 5, and so on. And yet, the way end users experience web *navigation* "macrostructurally" (Mata Pastor, pp.200-202), "hyperstructurally" (Torres del Rey and Rodríguez V. de Aldana 2014) or "superstructurally" (Jiménez Crespo 2013, pp.92-94) still revolves around concepts such as webpages as units, hyperlinks and forms as the main functional devices, and document tree structures stemming from a homepage and branching out through a series of sections and subsections.

It is on the server side where the main revolution —as far as developers and webmasters, but also translators and localisers are concerned— has taken place. To continue with Esselink's words: "Where translators could get started quickly by just working in Word or importing the *document* into a *translation memory system*, now often a localization *engineer* is needed to produce a 'translation *kit*' from a series of *complex SGML or XML files* containing the *manual text*" (Esselink 2002, p.5. Emphasis added). It is here, where the notions of whole documents and of straightforward import-export processes via translation memory systems are being challenged, that we decided to focus our research, spurred by the needs of our undergraduate localisation course at the University of Salamanca.

2. Motivation and nature of our research

Our main interest in new localisation processes for

dynamic webpages started in 2008, when we were asked to translate our Faculty's site from Spanish into our other working languages, with the collaboration of students. The website had been built with Joomla! 1.5, and was later made multilingual with the Joom!Fish extension[1]. We were given editing rights to this component, which allowed us to search for web articles, menus and other translatable elements, and to write or paste translated HTML content onto the editor window. In order to replicate the localisation process used for static HTML websites, client-side webpages were saved as only-HTML files, translated by means of a CAT tool with the aid of translation memories, terminology management and other integrated utilities, and the resulting HTML content was pasted onto the appropriate Joom!Fish editing environments.

Very soon, we decided we wanted to further explore how dynamic website localisation processes could be made more translator-friendly and, at the same time, to integrate them as seamlessly as possible with the whole development and publication cycle. However, the main drive behind this was to be able to analyse, understand and explain this evolving infrastructure and the new localisation needs and opportunities in order to enhance our localisation course, where students had learnt advanced website localisation concepts and strategies such as essential file types, languages and technologies, website (super, macro, micro and hyper) structures, including directory organisation and hyperlink types, folder structure cloning and hyperlink management, automation strategies (search/replace with or without regular expressions), etc.

In 2012, Joomfish was no longer available for the newest Joomla! 2.5, so we moved to the Joom!Fish-fork extension FaLang[2], which was also compatible with the more recent Joomla! 3.x version. It is important to note that the main goal of these third-party *internationalisation* extensions is to easily and automatically duplicate, for each newly activated site language, the monolingual web structure created with Joomla!, and to enable the editing and publication of translated content in all non-native languages. However the process of *localisation per se* was only facilitated when modules were created for the export/import (*interchange*) of translatable data and, particularly, when both technologies started to be merged: multilingual management and localisation interchange tools[3].

For the purposes of our research and, particularly, for our teaching practice, we adopted a twofold strategy: to try and generalise common features in the localisation of CMS-based websites and to illustrate this general process by setting up appropriate mechanisms and procedures to experiment it. We started looking at the architecture of other CMSs such as Drupal or Wordpress and the way internationalisation and localisation extensions were integrated. At the same time, we started developing an experimental tool that allowed us to automate a roundtrip export/import workflow for the Joomla! CMS we used for our teaching, and to identify and describe the main concepts, processes and possible breakdowns for translators' and localisers' work. This article mainly deals with our experiments in this process, particularly with the tool we developed for our teaching, the comparison with other available tools, and some conclusions regarding the general process of localisation and suggested basic translator and localiser needs.

One of the crucial questions in our roundtrip between the CMS and localisers' workstations was the format in which the interchange would take place, so, naturally, we looked at the possibilities offered by the two main standards in our field: the W3C Internationalisation Tag Set (ITS) and the OASIS XML Localisation Interchange File Format (XLIFF). While versions 2.0 of both standards will undoubtedly offer many new advantages to this process, at the time of our experiments and of writing this article they were still in draft status, so we used the latest approved versions, ITS 1.0 (W3C 2007a) and XLIFF 1.2 (OASIS 2008).

We have already introduced the three main components of our localisation research focus: 1. the product and the underlying technology relevant to localisation; 2. the interchange format for localisers to process; 3. relevant translation-oriented technologies for the processing of localisable texts, notably CAT tools. Our approach, as mentioned earlier, has to do with the integration of all three components in a way that is translator- or localiser- oriented, since it is these professionals who are in the best position to account for the intercultural task of negotiating the meanings, purposes, expectations and conventions which come into contact (and often into conflict) in the process of localisation, and to interpret objects, texts and meanings not for their own sake but for other people, users (at both –or the multiple– ends of communication), for whom translation and localisation is performed, for whom the translator is ethically responsible, and which determine meaning and transformations (Melby 1995, pp.122-132). However, as advocated in our localisation courses, in order for localisers to claim this expert position, they

must acquire a basic knowledge of the nature and mechanics of dynamic, CMS-based websites, particularly in so far as the technology influences both the communication production assemblage and localisers' own place in the development cycle.

In this regard, we feel that it is our duty to contribute to the reversal of the current wave of disempowerment for website localisers as we move from static to dynamic websites. The three components mentioned earlier empowered localisers working with static websites by providing them with (Torres del Rey and Rodríguez V. de Aldana 2011, 2014):

1 a high degree of visual and functional context;
2 specialised productivity, QA-performing tools;
3 the possibility of taking over engineering tasks for the multilingual restructuring of the overall website;
4 the possibility of delivering publication-ready directories and files.

CMSs have made the editing of individual articles within webpages and of interface items across the website easier. This has provided some visual content by allowing for (limited) in-context translation of articles. However, translators need to process the texts in the web product by means of their own tools in order to take advantage of the consistency, analysis, quality-check and terminology extraction functions (among others) built into them, and, often, to exchange the textual elements with other collaborators, who may use different tools. In CMSs, however, texts are still very much "locked" into databases. Localisers would also need write-access rights to the database and a multilingual component installed in order to try and recover some of the possibilities 3 and 4 above.

Even though automation seems to make the whole multilingual generation and publication easier and less error prone, any new web technologies and content management systems affect the way content is created and signifies. Localisers, as techno-linguistic experts, should not be nudged aside. This deskilling perspective, called the "idiot-proofing myth" by Adler and Winograd, "is more concerned with how to keep operators from creating errors than with enabling operators to deal with the inevitable contingencies of the work process". However, translation and localisation are all about dealing and negotiating with (linguistic, cultural, technological, contextual) dependencies, so we had better rise to the "*usability challenge*" (emphasis in the original) of making new technologies more effective by augmenting rather than replacing skills of localisers, by making the most of them (Adler and Winograd 1992, p.3).

3. Our experimental research

3.1 Overview

As indicated earlier, our main research goal was to provide localisation students with the conceptual and methodological tools to experiment with the process of localising a CMS-based dynamic website, and to do it on the basis of the three basic components for this task, i.e. the product and its technology; the interchange format; and the CAT tool. As we looked into the way these could be integrated, we also expected to draw insightful lessons for a translator-oriented approach to the task at hand.

It is only very recently that XLIFF extraction/merge tools have started being developed for Joomla! That is the main reason why we decided to build our own software, with the main purpose of experimenting with the data that needed to be exported and the way XLIFF could accommodate such data and the whole localisation process. Our tool was based on the multilingual extension to Joomla! developed by FaLang. As the extraction and merge operations were mainly made on the database tables created and managed by this plugin, we called our software FaLang2XLIFF.

FaLang2XLIFF[4] has proved very useful for our purposes, particularly considering our scarce resources, both economically and in terms of our available time. However, it has some limitations that need to be taken into account. Most importantly, it has been written in Java and is a stand-alone application which, unlike other related tools, is not embedded into the CMS as a module. Although this would make it potentially applicable to other database structures, both for Joomla! or other CMSs, it would also need to be given access rights to the database or to be run in the relevant network security zone or in a localhost.

Our application uses Schnabel's XSL stylesheet from his XLIFF Roundtrip tool[5], which converts XML files into XLIFF and back again. Drupal's XLIFF Tools is also based on that XSL file, so we briefly analysed its extraction performance. However, the alternative tool that we tested the most was JDiction[6], a multilingual management extension for Joomla! 2.5 which added an XLIFF extraction/merge tool in March 2013. Before describing the workflow of FaLang2XLIFF we will present some relevant conclusions from our analysis of JDiction.

All these tools, as well as our own, only deal with

content stored in the CMS database, i.e. articles or pages, modules (for instance, text in an add-on calendar), categories and other small user interaction elements, such as weblinks. The three main database elements (stored in text fields) for these contents are exported: web structure or interface (In Price and Price's terminology, cited in Jiménez Crespo 2013, p.58]) elements, longer (X)HTML article contents, and the technical parameters for the above elements. At this point, we have not yet considered processing dependent or linked files. Neither have we looked at CMS administration or content editor interface text, typically inserted in active pages (such as PHP, ASP or JSP) or externalised to text files (e.g. INI, PO).

3.2 Other extraction strategies

Our tests with JDiction for Joomla! 2.5 revealed certain problems for the localisation process. To start with, this tool shares with other multilingual managers, such as the current FaLang version, a shortcoming in that the article that is cloned for the target language cannot be edited in-context, on the webpage itself. Instead, the translation must be inserted in a separate environment, where the original text is not shown in parallel[7].

If we look at JDiction's XLIFF extraction/merge tool, we can appreciate considerable room for improvement too. For instance, items cannot be selected and exported individually; besides, this bulk process is applied on any one content type (categories, article contents, menus, menu items and modules) indiscriminately, regardless of whether individual elements are new, updated or they have previously been translated and approved. Finally in our analysis, all titles and article content receive "needs-translation" state values, irrespective of their actual status. On the other hand, parameters are always marked as "translated" in JDiction ("final" in Drupal's XLIFF Tools), which may cause the processing translation tool to edit unlocalisable values, resulting in the corruption of the database.

What is more, all extracted elements are lodged inside CDATA sections (Savourel 2001, pp.229, 298), which would commonly prevent parsing of the (X)HTML structure and segmentation based on it. This makes the XLIFF export no different from its pre-existing CSV export in JDiction. When filtered into CAT tools, articles would be processed whole, and tags would be mingled with actual text and unprotected. Even if, as Virtaal does by means of regular expressions such as <[^>]+/?>, tags are visually marked, translators would have a hard time trying to mentally reconstruct texts, identify and separate subtexts such as *alt* values, or correctly assign functional or layout tags to the appropriate words, phrases, sentences or paragraphs. Not to mention the high risk of messing with the code that this behaviour would entail.

CAT tools might alleviate the latter problem by integrating a WYSWYG editor for HTML content, which would also be triggered whenever the XLIFF *datatype* attribute value is "htmlbody", allowing users to switch between raw source HTML text and the visual representation of HTML tags on text. However, one thing that must be taken into account with CAT tools is that their XML filters are not always versatile enough and too often they only allow for the use of regular expressions to further filter translatable content, internal or external tags, and so on. In fact, XHTML should be processed with XML processors (e.g. XPath processors), in order to help interpret meaningful structures, which would produce shorter segmentation for better matches and translation memory leverage. This problem could be more easily solved, nonetheless, if the XLIFF export was carried out as HTML text and tags rather than plain text, or if, previously, the database could actually manage XML structures, as we will see later (see note no. 9).

title	introtext	metakey
Administrador de componen...	<p>Todos los componentes se utilizan también en el...	
Modulo Archivo	<p>Este módulo muestra una lista de los meses del ...	modules, content

Figure 1. An example of the three main database elements: title (structure), introtext (content) and metakey (parameter).

3.3 Workflow of our experimental application

Before using our tool, elements should be prepared for translation by means of FaLang's administrative interface (step 1). Currently, there are two important disadvantages in the behaviour of this extension: elements need to be selected and opened one by one, and the relevant source content must be copied onto the target content window.

Once target elements have been created for translation, it is the turn for our tool to connect to the database (step 2), for which it is necessary to provide the machine server name, the communication port (usually, 3306 for standard TCP/IP connections), the user ID having administrative rights, the user

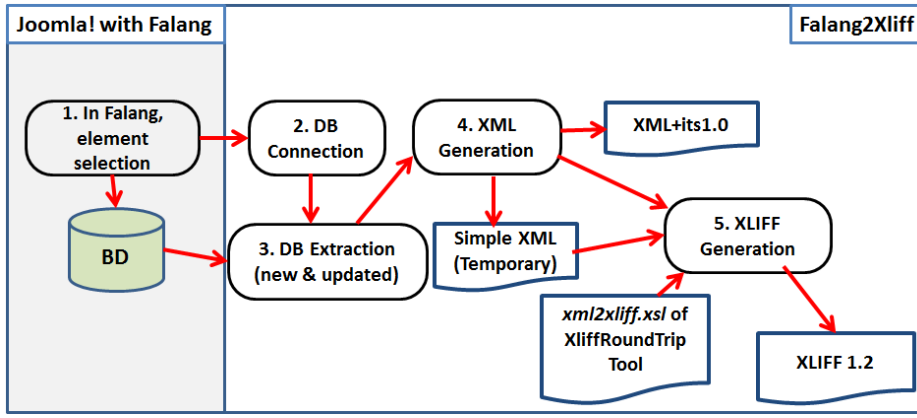

Figure 2. FaLang2XLIFF Workflow.

password, the name of the Joomla! database and the prefix or alias typically added to Joomla! table names.

Our application queries FaLang tables (step 3) but also the original content tables so as to check which data is new (i.e. established as translatable by the project manager by using the "Copy Source" procedure described in the first paragraph of this subsection) and which has been modified or updated in the source website[8]. In order to identify both types of data (new and modified), the MD5 hash code of both the translation record in the FaLang table and the corresponding record in the original content table are compared. Updates are identified whenever the source and target hash codes differ. On the other hand, translation content is considered as "new" when two conditions are met: source and target hash codes are the same and the "published" field of the FaLang record equals "0", i.e., it has never been published before, as otherwise it might mean that the source content has consciously been transferred to the target record (e.g. in the case of some proper nouns, trade names, etc.). Once new and modified translatable data are identified, their structural (e.g. titles) and content elements are extracted, but not technical parameters, as editing them may corrupt the database. However, in the future we will further analyse extractable parameters, as they may provide important contextual information for the localiser.

It is important to mention that the Joomla! HTML editor would have rewritten HTML fragments typed by users as XHTML (i.e. as correct XML)[9]. Nonetheless, our tool uses Jericho HTML Parser to recheck it and then rewrites data if necessary to make sure restricted characters in XML are escaped with their corresponding predefined character entity references (e.g. *&* for the & ampersand characters) (Savourel 2011, pp.44-47), attribute quotes are closed, and node hierarchies are kept. A current limitation is that all unpaired tags found in the XML hierarchy would be changed by our tool to self-closing tags without further analysis.

At this point (step 4), we would generate both an XML file with ITS rules and a temporary "simple" XML file that would serve as the basis for conversion to XLIFF by means of Schnabel's XLIFF Roundtrip XSL. It is worth noting that while intro and full texts are stored as HTML (<tags> and text) in the database, title fields contain only plain text and that no HTML/XML entities are processed. This means that we need to convert single characters, such as the ampersand, that may appear in the title field to their corresponding entity (*&* in this case) when processing the XML files, and then back to the single character when importing back to the database.

The XLIFF 1.2 file is successfully created (step 5) via the simple XML file just mentioned: here, database records are exported with <records_falang> as the root node and two child tags (see Fig. 3): <record_falang> carries, in its attributes, the administrative data that are needed to be merged back into the database; as a child node of the latter, <value_falang> contains translatable text, including HTML tags. We have adapted Schnabel's xml2xliff.xsl file used for the conversion so that the source language is variable (by using the XPath expression *{./@xml:lang}* as the value of the *source-language* attribute of the root element <file>) rather than just English ("en"). For clarity's sake, we have abbreviated some of the illustrated code by means of the ellipsis symbol "(…)".

However, we encountered several difficulties in the processing of the XLIFF file in CAT tools, as we will discuss later on. For that reason, after considering what might solve the problems we had identified, we

```
<?xml version="1.0" encoding="utf-8" ?>
<records_falang xml:lang= "es" host="..." db="..." alias="..." >
    <record_falang id = "453" (...) original_value ="3de7..."
                                    type="modificado">
        <value_falang><p><img (...)/>Usted tiene...
        </p></value_falang>
    </record_falang>
    <record_falang id="..." (...)>
        <value_falang>Usando Joomla! &amp;amp; Componentes
        </value_falang>
    </record_falang>
</records_falang>
```

Figure 3. XML file generated by FaLang2XLIFF.

decided to produce a second version of the "simple" XML file described earlier, injecting it with ITS 1.0 rules regarding the processing of translatable elements and of segmentation-related text-element relationships (W3C 2008), as follows (see Fig. 4):

- We used global (not local) rules, directly embedded in the resulting XML file.
- All <value_falang> nodes and their child nodes were made translatable; all other nodes are not translatable[10].
- Within <value_falang> nodes, HTML attributes typically carrying text are made translatable. Href attributes are also localisable when they start with "http://" or "https://" (i.e., generally, when they are external site references) [11].
- HTML elements that can occur inside text sentences (such as <a> or) are considered Within Text, which prevents segmentation (see later).

The return trip to the database is also performed by FaLang2XLIFF, so far irrespective of translation status (e.g. nodes marked as "needs-translation" will still be imported back to the database). Again, a temporary XML file needs to be produced from the XLIFF 1.2 file before SQL generation. The database can be updated directly online, although an SQL file will also be produced, in case the update is to be done in batch mode.

3.4 Analysis and Discussion: interchange problems

The application of general-purpose XSL transformation files to specific mark-up languages such as XHTML when written by CMS HTML editors may present a series of limitations. One of the consequences of this is that attribute values would not be extracted to XLIFF <trans-unit> elements in XLIFF. Take, for instance, the Joomla! article in Fig. 5, with the following source code:

```
<?xml version="1.0" encoding="utf-8" ?>
<records_falang xml:lang= "es" host="(...)" db="(...)" alias="(...)"
 xmlns:its="http://www.w3.org/2005/11/its" its:version="1.0">
<its:rules xmlns:its="http://www.w3.org/2005/11/its" version="1.0">
<its:translateRule selector="//*" translate="no"/>
<its:translateRule selector="//value_falang//* | //value_falang"
translate ="yes"/>
<!--translatables attributes-->
<its:translateRule selector="//value_falang//@alt | (...) |
//value_falang//@href[starts-with(.,'http://') or
starts-with(.,'https://')] |
 (...)" translate="yes"/>
<!--Elements within text-->
<its:withinTextRule selector="//value_falang//a | (...) |
//value_falang//img | //value_falang//span |(...)" withinText="yes"/>
</its:rules>
<record_falang id ="...">
    <value_falang>(...)</value_falang>
</record_falang>
</records_falang>
```

Figure 4. ITS rules injected in the output XML file. Ellipsis (…) is used.

- ** **<*img alt= "Joomla! Spanish, versión 2.5"*>Usted tiene un sitio Joomla! 2.5 adaptado y traducido por Joomla! Spanish** **
- ****Joomla! es de <*a href= "http://opensource.org" title= "Iniciativa Open Source"*>código abierto</*a*>. Joomla! hace que sea:****
- ****Fácil <*strong*>crear</*strong*> y construir un sitio web de manera que quiera.****
- ****Bastante <*strong*>sencillo</*strong*> de <*strong*><*em*>actualizar y mantener.</*em*></*strong*>****

The XliffRoundTrip transformations to XLIFF 1.2 are as follows:

- tags without text are included in <group> elements (html tags without text; highlighted in grey in our source code);
- tags with text are inserted in <trans-unit> elements (in **bold**);
- inline or within text tags are transformed into <g> </g> pairs or into <x/> xliff elements (in italics).

This would allow us to localise the *alt*, *title* and *href* attribute values (provided the latter starts with http:// or https://). We could also transform this XML file to XLIFF successfully by means of Okapi Rainbow, which also supports global *Translate*, *Elements Within Text* and *LocNote* ITS rules (W3C 2007b), and then import it into any XLIFF-supporting CAT tool.

Another problem that could be averted with ITS rules has to do with HTML overtagging, typically produced by CMS HTML editors. If, for instance, we are writing the above article in the CMS editor and we undo the list item or the whole ordered list and then change the paragraph configuration, Joomla! would add pairs of tags with style attributes around paired , or <a> tags including text. According to the transformation rules for XLIFF indicated earlier, that would produce undesired oversegmentation, as text within new paired elements would be included in their own <trans-unit> nodes (i.e. in independent segments or translation units). To sum up, many reformatting actions on the CMS html editor cause html overtagging, which can hardly be safely undone by CMS Clean-html functions.

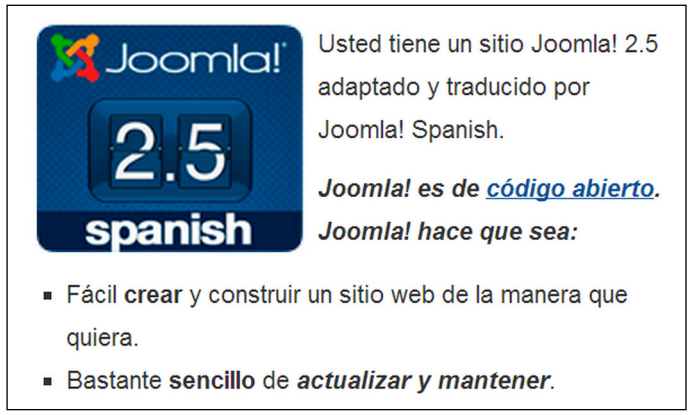

Figure 5. Sample Joomla! article.

The resulting XLIFF file would not include translatable attribute values within <trans-unit> nodes. Instead, inline tags would have an id reference to said values, which would be kept in the skeleton part of the file for later merging.

However, the XML file with ITS rules that Falang2XLIFF generates would be processed more effectively by a CAT tool – such as SDL Trados Studio– that does support global and embedded ITS rules for features *Translate* and *Elements Within Text*.

4. Towards translator-oriented localisation of CMS-based websites

Solutions to the internationalisation and localisation of CMS-based websites tend to focus rather narrowly on the technical aspects related to the extraction/merge roundtrip of translatable data or on the user-friendliness of integrated multilingual management and in-context article edition. However, little or no attention is paid to the overall communication needs that translators and localisers

must address in order to do their job successfully from the point of view of the pragmatic, intercultural, interlinguistic exchange that they are commissioned to perform.

It is true that the technical solutions mentioned earlier bring the localisation process a step closer to human, translation-oriented concerns:

- by using an XLIFF file, translation data can be enriched with information on the localisation process, objects, intentions, and so on;
- by partaking in the roundtrip, the localiser may not be seen as a "dysfunctional" agent in the technological process, but as an "enabler" in the infrastructure of (multilingual) content management;
- by handling a standard interchange format, localisers can use their computer-assisted tools and benefit from translation memories, terminologies, quality control, and all other integrated translation aids;
- alternatively, by being provided a simple system to enable and manage the multilingual structure of the website, they can devote more attention to translation matters, including negotiating contrastive conventions of web genres (Jiménez Crespo 2013, Ch.4);
- finally, by allowing localisers to place (and replace) translations in allocated webpage spaces (for some CMS content types), they benefit from a more contextualised approach to the translation of web articles.

However, the above advantages are currently far from being fully realised, particularly the combination of in-context visual translation and XLIFF support (which also show some limitations, as we have seen in the previous section). In general, a holistic view is missing as regards the part that the different technological, textual and semiotic components play in the task of the localiser, and how they can be realised in the translation process.

A translator or localiser is an intercultural mediator who makes sense of a text (or an interrelated series of texts)[12] that has been produced in a specific cultural, professional and technological context, and creates a version of that text in a different human language, taking into account differences between source and target contexts, the (explicit or assumed) purpose of the textual exchange and foreseen effects of the resulting text (in the target system or context, but also as regards the source context of production), and formal or informal norms and conventions regulating translation and localisation, usually related to culture-bound ideas of equivalence, adequacy, comparativeness and functional adaptation.

All these contextual, cultural, technological, purpose-bound considerations have a huge impact on the task of the localiser, just as they implicitly or explicitly condition the original text production process. The intercultural mediator, furthermore, needs to stand astride (or to constantly move across and back) the source and the target cultures and language systems, and to make informed decisions in order to communicate or negotiate global and particular meanings, functions, intertextual relations, purposes and (intended or unintended) effects which have been formed, structured and expressed in a linguistic mould and in a cultural context which can never be symmetric or equivalent with the target language and culture.

The other major communication, sign-producing system that greatly influences the production of meaning is the technological – here, the website as a product and the CMS as an agent mediating structure, communication, document or text boundaries, and, in general, the interaction of knowledge between users (designers, contributors, consumers, "browsers-by", and so on), the web genre and the information to be displayed.

However, web CMSs tend to gear their modus operandi towards monolingual, monocultural production, not only because multilingual management extensions are often a later add-on (and not as user-friendly or flexible as the interface for original content editing), but also because there is a source-oriented inherent assumption of direct, objective, unproblematic, ungrounded semantic correspondence (Winograd and Flores 1986, p.18; Melby 1995, pp.122-132) between the genesis of meaning and intention and the infrastructure and applications enabling and conditioning their expression. This kind of correspondence is circular (meaning > production structure, materials and mechanisms [language, writing and technological systems] > meaning adjustment > system adjustments > meaning, etc.) and is not recreated and rarely unveiled for localisation. Thus, shockingly enough, localisation tends to be left out of the meaning-production cycle.

Localisers are usually provided small chunks of text (either in the CMS editing environment or as translation units in bilingual interchange files) for

ease of exchange and integration in the localisation or publication technologies. Even if "experts most likely develop strategies either in a pre-translation stage (by acquiring prior knowledge of the global hypertext […]), or during the translation process ([…] from a prototypical of the digital genre in question and [by] negotiating the macro and microstructural levels) to compensate for the lack of context" (Jiménez Crespo 2013, p.64), localisation efficiency can be severely disrupted by forcing constant negotiation between meaning-structure levels, context recreation, and, particularly when "translating interaction", i.e. when texts and messages are not on the immediate surface visible webpage layer.

Any web content is meaningfully integrated in a larger information unit (e.g. a bigger article or a web page), next to other subunits (or subgenres), and also within a larger whole (the website, or even the World Wide Web). Localisers usually receive only the small subunits, with little or no information of relative position, order or functional dependencies. However, these units are coherently and cohesively (Jiménez Crespo 2013, pp.59-62) inserted (at least) in:

- the more general or particular communicative or performative functions they are part of;
- the regions or positions they appear in, which also have communicative or semiotic significance;
- the hypertextual, interactive relationships they are part of or which they include;
- macrostructural relationships (e.g. the particular location in the sitemap or the order they appear within a group or element such as a menu);
- the conventions for the type of element they are in (a more or less ephemeral article, a more stable basic page, a module, a category classifying blog entries, etc.);
- potentially indexed search results.

In this regard, CAT tools need to be able to offer relevant contextual information to prevent localisers from concentrating exclusively on the microtextual level (Jiménez Crespo 2008, pp.5-6), and for this, XLIFF development and CAT support (and visualisation and interaction) of this interchange format must grow closer together. However attractive, in-context translation in the CMS editing environment without the benefits of Computer-Aided Translation Technology can be dangerously insufficient as it would not profit from some of the main benefits of CAT technology, if used properly:

- translation and terminological consistency, particularly as regards specialised knowledge, web genre conventions, brand or client-related phraseology and terminology, and so on;
- quality checks;
- productivity functions;
- filtering and transferring format and presentation;
- language/knowledge building and annotation;
- team work and exchange functionality, particularly as large websites tend to be localised by more than one professional.

An important step forward would be for web CMSs to incorporate localisers and localisation into their content management agents, definitions and mechanisms, since the amount of content that localisation handles and transforms is substantial. One way to do this is to support and encourage the generation of ITS 2.0 (W3C 2013) metadata for translatable elements and attributes, text analysis (content, structure, relations between parts), external resources (e.g. relevant intertextual, intermedia references, whether explicit or implicit, that are important for overall meaning construction), size or other restrictions, linguistic annotation and any other features that may affect data interchange (via XLIFF 2.0 [OASIS 2013]).

Another complementary way would be to provide the appropriate mechanisms for a localisation project manager (PM) profile in CMSs. This user would be able to annotate content and include relevant metadata (e.g. specific localisable external links, localisation notes, text analysis, and so on), or prepare localisation interchange files, by grouping translatable content with contextual non-translatable content, including an html preview skeleton file, linking appropriate XSL/CSS files for better visual contextualisation, or, simply, providing URL links for each group of localisable content.

5. Conclusions and future work

The declared purpose of CMSs is managing content in a structured, knowledge-sensitive (and sensible) way. Localisation should therefore be part of their core concerns, and it would be sensible if CMSs integrated the Internationalization Tag Set with their content generation strategies, and XLIFF with their multilingual content interchange mechanisms.

Particularly, localisation of whole or large sections of websites (as opposed to periodic translation of individual, more-or-less independent articles) and

web localisation training would greatly benefit from textual signposting and contextualisation strategies, which could be included in internationalisation metadata for the original content (by authors or localisation PMs) and transferred or enriched in the XLIFF files that localisers would process with their CAT tools.

This is one of the avenues of experimental research that we will pursue in the near future: the extraction of contextual information that can be useful for CMS website localisers and can be integrated in XLIFF files for CAT work. At the same time, we will intensify our analysis of other roundtrip tools (and other possible localisation strategies) for web CMSs, and the way these content management systems design their interaction with web objects, concepts, conventions, meaning and interrelationships. Finally, we will continue to look into CAT integration of current and future CMS-based web localisation processes.

After all, we need to understand the way information, knowledge and communication is conditioned and shaped by technology (expanding some possibilities, reducing others, creating new meanings) in order to try and reach an understanding between the different professional languages involved in dynamic web localisation, to build (by assimilation, contact, translation, etc.) common metaphors that may help translators and localisers (and their trainers) to "inscribe" translation values and meanings in the operating system of CMS technologies (Torres del Rey 2005, pp.105,121-134), often by means of standard languages.

Acknowledgements

The work presented here has been carried out in the framework of the research project "Regulación de los procesos neológicos y los neologismos en las áreas de neurociencias" (FFI2012-34596), which receives funding from the Spanish Ministry of Economy and Competitiveness.

References

Adler, P.S. and Winograd, T. (1992) 'The Usability Challenge', in Adler, P.S. and Winograd, T., eds., *Usability. Turning Technologies into Tools*, New York & Oxford: Oxford University Press.

Esselink, B. (2002) 'Localization Engineering: The Dream Job?', *Tradumàtica*, (1), available: http://www.fti.uab.es/tradumatica/revista/articles/besselink/art.htm [accessed 24 Oct 2013].

ISO/IEC (2011) *Information technology — Database languages — SQL — Part 14: XML-Related Specifications (SQL/XML)*, 9075-14:2011 [online], available: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=53686 [accessed 24 Oct 2013].

Jiménez Crespo, M.A. (2008) *El proceso de localización web: estudio comparativo de un corpus comparable del género* sitio web corporativo, Ph.D. dissertation, Granada: Universidad de Granada.

Jiménez Crespo, M.A. (2013) *Translation and Web Localization*, London & New York: Routledge.

Mata Pastor, M. (2005) 'Localización y traducción de contenido web' in Reineke, D., ed., *Traducción y localización: mercado, gestión y tecnologías*, Las Palmas de Gran Canaria: Anroart, 187-252.

Melby, A.K. (1995) *The Possibility of Language: a discussion of the nature of language with implications for human and machine translation*, Amsterdam/Philadelphia: John Benjamins.

OASIS (2008) *XLIFF 1.2 Specification.* OASIS Standard 1 February 2008, available: http://docs.oasis-open.org/xliff/xliff-core/xliff-core.html [accessed 24 Oct 2013].

OASIS (2013) *OASIS XLIFF Wiki Frontpage*, available: https://wiki.oasis-open.org/xliff/ [accessed 24 Oct 2013].

Savourel, Y. (2001) *XML Internationalization and Localization*, Indianapolis: Sams.

Torres del Rey, J. (2005) *La interfaz de la traducción: formación de traductores y nuevas tecnologías*, Granada: Comares.

Torres del Rey, J. and Rodríguez V. de Aldana, E. (2011) 'La localización de webs dinámicas: presente y futuro', accepted for *1st International T3L Conference*, Universitat Autònoma de Barcelona, June.

Torres del Rey, J. and Rodríguez V. de Aldana, E. (2014, forthcoming) 'La localización de webs dinámica: objetos, métodos, presente y futuro', *JoSTrans*, (21), available: http://www.jostrans.org.

W3C (2007a) *Internationalization Tag Set (ITS) Version 1.0*, W3C Recommendation 03 April 2007 [online], available: http://www.w3.org/TR/its [accessed 24 Oct 2013].

W3C (2007b) 'Test Suite', in *Internationalization*

Tag Set Version 1.0, Version: $Id: Overview.html,v 1.56 2007/02/27 06:20:03 fsasaki Exp $, available: http://www.w3.org/International/its/tests/Overview.html [accessed 24 Oct 2013].

W3C (2008) '5.1.4. Associating existing XHTML markup with ITS', *Best Practices for XML Internationalization*, W3C Working Group Note 13 February 2008 [online], available: http://www.w3.org/TR/xml-i18n-bp/#relating-its-plus-xhtml [accessed 24 Oct 2013].

W3C (2013) *Internationalization Tag Set (ITS) Version 2.0*, W3C Proposed Recommendation 24 September 2013 [online], available: http://www.w3.org/TR/its20 [accessed 24 Oct 2013].

Winograd, T. and Flores, F. (1986) *Understanding Computers and Cognition: A New Foundation for Design*, Norwood (NJ): Ablex.

Notes

1 http://www.joomfish.net/

2 http://extensions.joomla.org/extensions/languages/multi-lingual-content/18210.

3 Josetta (http://anything-digital.com/josetta/) is another multilingual manager for Joomla. XLIFF Tools (https://drupal.org/project/xliff) is both a multilingual manager and an XLIFF roundtrip tool for Drupal, just like WPLM (http://wpml.org) for Wordpress.

4 http://diarium.usal.es/codex/desarrollo.

5 http://sourceforge.net/projects/xliffroundtrip.

6 http://jdiction.org.

7 In the case of FaLang, this seems to be a bug, as the editing window for the target language does work (and with the original text visible) but the result is inserted in the native language tables.

8 For an analysis of the database tables and attributes that are queried, see our article (Torres del Rey and Rodríguez V. de Aldana 2014). As mentioned earlier, an in-depth analysis of the way other CMSs (or their multilingual managers) organise tables and translatable elements would allow us to extend the functionality beyond Joomla! with FaLang.

9 XHTML elements should be stored in databases as XMLElements, as recommended in ISO/IEC (2011). Unfortunately, at the moment XML support is low in MySQL, which is the favoured database management system for CMSs. We believe that it would be beneficial to adopt other systems with more advanced XML functions such as PostgreSQL or to press for further XMLsupport in MySQL.

10 "In case of conflicts between global selections via multiple rule elements, the last selector has higher precedence" (W3C 2007a, Sec. 5.4).

11 ITS 1.0 supports XPath 1.0, which does not support regular expressions, which would have made a few conditions simpler than with Xpath syntax.

12 For the purposes of this article, "text" also means hypertext and associated multimedia and interaction. All technical adaptations that may be necessary in the localisation process are considered as part of the interpretation of the text as we have just defined, and will not be covered here mainly because our focus is on the export/import of textual material from CMS-based websites.