[From: *Equivalences* 32/1, 2005]

Coping with the time-to-market factor in software localisation projects

Federico Gaspari

Centre for Computational Linguistics
University of Manchester Institute
of Science and Technology (UMIST)

Abstract

This paper discusses the importance of using computer-assisted translation (CAT) tools in software localisation projects, and sheds light on their key importance for the success of translators who are involved in the localisation industry. The main focus is on the vital role played by tools such as translation memory (TM) systems end terminology management packages to help translators cope with the pressure exerted by the time-to-market factor in real-life scenarios. The paper provides some background information about localisation in general, and presents in more detail a number of practical issues that illustrate the crucial impact of CAT tools on software localisation projects. Finally, the importance of project management in localisation-oriented processes is emphasised, and some new emerging working practices for translators are outlined.

Résumé

Les outils d'aide à la traduction sont aujourd'hui largement utilisés dans les projets de localisation. Les mémoires de traduction et les SGBD terminologiques, en particulier, permettent aux traducteurs de faire face aux délais serrés exigés dans des projets de traductions réalistes. Cet article explique ce qu'est la localisation de logiciels et le rôle qu'y jouent les outils d'aide à la traduction. Il aborde en finale la gestion de projets et les modifications des pratiques professionnelles que ces nouvelles pratiques ont entrainées.

1. Introduction

1.1. Overview

This paper looks at software localisation and discusses the main issues connected with the so-called time-to-market factor. By providing insights into how software localisation projects are managed and carried through, the main emphasis is laid on some of the requirements that are bound to give professional translators involved in the competitive arena of localisation a lead in the field.

This discussion is focused on the localisation of software products, such as operating systems, software programs and plug-in applications. In reality, localisation is also associated with multimedia products (e.g. CD-ROMs with games and Interactive applications, etc.) and on-line material, such as the contents of web-sites and multimedia documents found on the Internet, since they require extensive and careful adaptation to be used by people in various areas of the world or with different cultural backgrounds. Due to its limited scope and length, though, this paper will only be concerned with the localisation of software products, which will be considered here by taking the time-to-market factor into account (a discussion from a similar perspective providing a specific company-based case study can be found in Topping 2000).

The software localisation industry began to develop in the early 1990s and is now a fast-growing field worldwide. In general, localisation is concerned with the complex process of adapting products to meet the linguistic, cultural and legal requirements of specific environments or target regional markets, the so-called locales (cf. Esselink 2000, Esselink 2003, O'Brien 1998, Thibodeau 2000 and Topping 2000). The notion of locale is more elusive than one may think, since it is quite distinct from the concept of country or state and goes beyond geo-political considerations; the notion will be explored in more depth in section 2.1. below.

Software localisation encompasses translation as one of its core components but is not limited to it, since it adds to the linguistic and cultural aspects a technical and a content dimension. This complex framework calls for continuous cooperation and an ongoing exchange of feedback and input between software engineers, developers, testers (who simulate the role and expectations of the intended end-users), clients, translators and their project managers. As a result, localisation projects are complex processes that need to be

carefully managed and supervised, and typically involve a variety of people with different backgrounds and skills, who rely on an array of tools and co-operate by combining their efforts to guarantee the quality and success of the end product.

1.2. Background

Professional translation has been significantly affected over the past decade or so by the use of Internet-based resources, web-enhanced working methods (see e.g. O'Hagan 1996 and O'Hagan & Ashworth 2002) and the so-alled computer-assisted translation (CAT) tools: translation memory systems, terminology extraction and management packages help human translators increase their throughput, maximise their productivity and streamline the multilingual documentation workflow, e.g. by speeding up turnaround times and enhancing consistency across related documents or similar projects (Austermuhl 2001: 102-123, 134-151, Bowker 2002: 92-127, Bowker 2003, Esselink 2000: 362-378, Gordon 1997, Heyn 1998, Hutchins 1998a, Hutchins 1998b, Kenny 1999, Merkel 1998, O'Brien 1998, Somers 2003a and Somers 2003b).

This technological evolution calls for an ongoing and regular update of the translators' own skills and working methods, if professional activity is to be successful and they wish to offer top-quality high-profile translation services. This general requirement becomes particularly essential for translators who work in software localisation projects, since the proficient use of state-of-the-art translation technology tools and familiarity with telecommunications and IT-related issues are bound to have a huge impact on their working practices.

2. The challenges of software localisation projects

2.1. Locales: regional markets around the globe

Some clarification is needed at this stage of the discussion to explain more precisely what locales are, owing to their key role in localisation - the very name localisation derives from the notion of locale. The main points to be considered When thinking of a locale are those of linguistic unity and cultural consistency, Which are not sufficient for a comprehensive description, however, as the following examples will show. In this respect, for instance, Switzerland is usually not considered a single entity, because of the different national languages that are officially spoken on its territory. As a result, in spite of being one country in

the geo-political sense, from this point of view Switzerland encompasses one locale for German (associated with the German-speaking countries), one for Italian (a virtual extension of the mainland of Italy) and one for French (associated with the other French-speaking communities in the rest of the world). Even though other minority languages are spoken by Swiss citizens or residents, their very restricted use or limited commercial interest makes them virtually irrelevant for localisation, and hence for the definition of a self-contained locale.

The Swiss example is a rather extreme but self-explanatory one. On the other hand, even though the official language in both Portugal and Brazil is Portuguese, strictly speaking these two countries represent two different locales, because of the differences between the European and Brazilian varieties of Portuguese. Similarly, the United States of America is usually considered a single locale together with the English-speaking portion of Canada, being kept separate, however, from the United Kingdom and Ireland most of the time. As a matter of fact, the English language used in North America differs from that In Europe (e.g. for preferred spelling conventions and use of some vocabulary); at the same time, cultural and legal differences combined with subtle preferences In taste (e.g. In terms of shared social and moral values, historical and cultural backgrounds, etc.) and common conventions (one may think of the different currencies and formats used for figures, dates, temperatures, etc.) are more noticeable between the speakers of English in the American and the European continent.

The few examples provided here should illustrate how fragmented locales are, since they are not necessarily characterised by geographical contiguity, and there is no one-to-one correspondence to national states. One further general point should be emphasised here, namely that the vast majority of localisation projects are carried out starting from the original 'native' version of a software product in English, most often American English. This is due to the fact that software houses and developers are typically based in the United States, or have privileged commercial partnerships with North American companies. As a result, the most considerable share of the worldwide software localisation business is represented by the adaptation of products originally designed and developed for (American) English-speaking users, which have to be made available for people from different linguistic and cultural backgrounds as well. Furthermore, target languages for localisation projects are usually selected among those with extensive populations of potential buyers and users

in the developed and wealthy areas and countries of the world, since they are likely to attract significant commercial or business interest.

2.2 Software localisation and internationalisation

Software localisation projects aim to transform a software product to make it more suitable for, or acceptable to, a group of target users that is different from the original one, i.e. for a new locale. These changes may be more or less extensive, depending on the stipulations and specifications that apply to each localisation project; these cannot be predetermined but are negotiated in every case by project managers according to clients' needs and requirements. In general, software localisation by definition requires the complete packages and all relevant support documentation to be translated into the language(s) used in the area(s) of new target markets and locales that a company, vendor or software manufacturer wishes to penetrate.

This translation process into multiple target languages involves not only paper manuals, printed user's guides, copyright contracts, manufacturer's guarantees, terms and conditions of purchase, then, but also tutorial demo sessions, menus, screen interfaces and commands, pop-up windows, installation wizards, context-sensitive help, on-line user guides, etc., that is to say all the elements that incorporate linguistic strings and textual information of some kind. This paper focuses on the issues connected with the multilingual localisation of software products which involves the translation of both accompanying documentation and the graphic user interface (GUI). The latter refers in particular to all the information that appears on the screen and through which the user interacts with the software, e.g. by giving instructions and commands to the software, so as to access functions and perform operations. Since they are not crucial to the viewpoint adopted here, considerations of a strictly technical nature relating to software engineering are kept in the background of the discussion, even though they do affect the time-to-market factor insofar as they have an impact on the translation process.

One general important point to bear in mind is that the adaptations that take place during localisation projects do not in principle improve or enhance the performance and functions of software packages. However, user-friendliness and usability are obviously increased if on-screen commands, manuals and all other support documents are available in the user's native language.

Similarly, an accurate localisation project covers other extra-linguistic features such as formats adopted for numbers and figures (e.g. decimals), dates, temperatures, currency, icons, etc., that are changed to those most familiar or customary in the target culture according to established conventions and general taste. These adaptations are ultimately aimed at maximising the potential business impact of exporting localised software products into several regional markets, so that local customers' needs are met in order to give the impression that a particular piece of software is suited to them: "A major reason for localizing software is economic. [...] A product that is barely making a profit domestically can be a highly profitable venture overseas. All else being equal, a software product that is not localized is less likely to survive over the long term." (Thibodeau 2000: 127).

Localisation is closely related to the key concept of internationalisation, which implies the process of engineering and designing a software product so that it can later be easily localised, thus encompassing a variety of considerations of different kinds (cf. Del Galdo & Nielsen 1996, Esselink 2000, Esselink 2003, Hall & Hudson 1997, Luong *et al.* 1995, Thibodeau 2000: 144-145 and Uren e*t al*, 1993). For instance, fields used to fill in on-screen forms and dialogue boxes with in-built textual contents on graphic user interfaces should be shaped and positioned so that they can accommodate strings of characters that are longer or arranged differently in a given language when compared to those of the original release. This normally happens when translating from English into some other European languages, into Asian languages with non-alphabetical writing systems or into languages with a right-to-left writing and reading sequence such as Arabic. The idea behind internationalisation is that the original software product should be designed and engineered taking into account that it may later be modified and localised, and steps need to be taken to ensure that adaptation of the GUI can take place smoothly and accurately.

So-called pseudo-localisation, for example, is the process of inserting random letters and characters into menus and dialogue boxes of a software product in a sort of pre-translation (usually with diacritics or language-specific symbols) to make sure that the original design of a software product allows for a variety of alphabets and scripts of different languages to be correctly displayed. Pseudo-localisation also makes it possible to test and check another crucial aspect of internationalisation, namely that all textual strings have been adequately externalised and that the layout of a GUI tolerates a certain expansion

rate. In other words, a 10% or 20% expansion rate for textual strings (e.g. options on menus or names of fields and buttons that appear on the screen) allows for translation into a target language that creates strings 10-20% longer than their counterparts in the original source language, in order to avoid cluttered text boxes or truncated strings in the GUI of the localised version.

Re-sizing strings of text, buttons, icons and dialogue boxes is very often necessary when localising graphic user interfaces, and there are professional software localisation toolkits that help deal with the various processes involved in these operations. Large software companies with significant international sales as well as single and multiple language vendors (see section 3.1. below) may use proprietary localisation tools and translation memory software developed in-house (cf. Esselink 2000: 377-378, 383-394 and Esselink 2003: 79-84), While other tools to handle the localisation of GUIs are also available on the market and can be purchased by translation agencies or freelancers (e.g. Pass Passolo and Alchemy Catalyst are popular ones[1]). These localisation kits can be Integrated with translation memory and terminology management tools (as well as with fully automatic machine translation systems) with a view to increasing Consistency and speeding up the overall translation process within the localisation project, thus helping to minimise the time-to-market (see sections 2.3., 2.4. and 3.2. below and also Austermühl 2001, Bowker 2002, Bowker 2003, Esselink 2000: 362-378, Esselink 2003, Heyn 1998, Merkel 1998, O'Brien 1998, Somers 2003a, Somers 2003b and Topping 2000).

Internationalisation policies usually encourage the adoption of culture-neutral and language-independent platforms for encoding linguistic information into operating systems and software applications. These international standards help to create environments that facilitate the smooth transition to multiple versions of the software package when it is to be localised into other languages. For instance, software developed according to ISO- or UNICODE-compliant encoding systems[2] in the original release (say, in American English) can be more easily rendered and adapted - at least as far as linguistic requirements are concerned - into other languages as diverse as Chinese, Japanese and Korean.

---

[1] Further Information and downloadable demos are available at the URL's http://www.passolo.com and http://www.alchemysoftware.ie - these web-sites have been accessed and are available on-line as of 20th August 2004.

[2] Further information is available at the URL's http://www.iso.ch and http://www.unicode.org - these web-sites have been accessed and are available on-line as of 20th August 2004.

Internationalisation issues, however, are not restricted to the graphic user interface on the screen, but also typically take into account the easy management and adaptation of images and information given in on-line and printed accompanying documentation, e.g. on CD-ROM or in paper-based reference material: in this respect, effective DTPoriented internationalisation may involve leaving enough space for larger captions under screen shots and figures, envisaging the presence of re-sizeable text boxes that explain parts of the user interface or draw the attention of readers to elements such as icons, buttons, pop-up windows, dialogue boxes, glyphs, etc. in the translated documentation materials.

2.3. Software localisation and time-to-market

The time-to-market factor is generally defined as the period elapsing between the moment in which the development of a product has been technically completed, thus making it potentially ready to be used and marketed, and its actual release to customers, i.e. when it is finally available for purchase. The notion of time-to-market is associated with that of cycle time, which covers a more extended period of time, however, as it is "the elapsed time from the beginning of idea generation when the firm decided to develop a new product to the end of product launch when the product is commercially available and managed in a routine manner" (Ali *et al.* 1995: 56).

There is a significant body of research investigating the crucial importance of the time-to-market factor on product development and performance from a variety of perspectives, given the fact that "time-to-market decisions clearly play an important role in determining the ultimate success or failure of a new product" (Bayus 1997: 485), and that "reducing cycle times in the new product development process has [...] become a critical objective for most companies" *(ibid.:* 486). As a result, in a variety of competitive areas such as manufacturing industry and other industrial sectors "time to market has moved from obscurity to a prominent topic among product developers" (Smith 1999: 222), and a number of studies have considered the trade-off that exists between accelerating time-to-market and compromising on quality when developing a new product (see, for example, Cohen *et al.* 1996 and Morgan *et al.* 2001).

For software packages that need to be localised before they can be sold in certain regional markets and locales, time-to-market duration depends heavily on the throughput time of translators, or on the time they require to finish their

job. As a result, professional translators are directly exposed to the pressure of extremely tight deadlines, and invariably the time frame tends to be reduced, even to complete long and complex multilingual localisation jobs: "Short time-to-market cycles for innovative products are critical for success in today's competitive business environment. [...] As long as the time spent on localization is greater than zero, there will always be pressure to do it faster, better, and more cost-effectively" (Topping 2000: 125).

Software products are typically sold with complete support documentation and all the linguistic interfaces translated into the desired language(s), depending on which local markets around the world are being targeted. In an increasingly global and competitive market it is of the utmost importance that the time-to-market be kept as short as possible: in the software localisation business in particular, the time factor is of crucial importance, and the translation processes and cycles involved benefit from adopting practices, methods and tools that allow teams of human translators to maximise, and to a certain extent also to standardise, their productivity.

2.4. Simultaneous release of localised products in regional markets around the world

Even though locales can be fragmented and scattered around the world with no territorial unity (as explained in section 2.1. above), it is usually the case that software vendors want to release and launch their new products simultaneously in several areas where they have marketing interests, possibly in all the regions that they wish to target at the same time. This synchronic strategy is aimed at maximising the benefits of launching a new product (e.g. an upgraded operating system, the new release of an already existing program or a brand new suite of applications) worldwide, so as to exploit the positive impact of global marketing and advertising campaigns to their fullest extent.

The simultaneous release (also called 'simship' - cf. Esselink 2000: 19, 111, 143) of a single new product worldwide is also likely to minimise the risk of illegal pirate copies of the software being sold at a lower price before the official product is released by the vendor. If, for instance, an original software package is sold in the United States first, and then released to an Asian locale in the relevant localised version only six months later, it could happen that by the time it (to presented in the foreign market illegal copies have already been circulated,

thus dramatically reducing the potential revenues and damaging the image of the manufacturer. When releasing a new (or upgraded) software product world-wide, simultaneous shipment to all targeted locales is a strategic choice that is bound to bring substantial competitive advantages over other vendors or manufacturers, especially those that may be interested in marketing similar products in the same geographical areas. Releasing a software product in a locale before one's competitors may substantially increase profits and guarantee a long-lasting leading position in sales and all related business activities (e.g. customer service, after-sales support, technical assistance, repair centres, etc.): "a localized version [of a piece of software] can extend a product's life cycle. If the domestic market Is declining, an international market may be emerging or still growing, and sales abroad will help finance the next generation of products" (Thibodeau 2000: 127).

These considerations apply to software products as well as to most other goods for which there is fierce competition on a global scale, as is testified by All e*t al.* (1995: 55): "Getting new products to market quickly is critical for attaining competitive advantage in the battle for global markets. [...] shorter times to develop new products (cycle times) can increase sales through extended product lives, Increase market share through pioneering, increase profitability [...], and enhance a company's image as an innovation leader." Reducing the time-to-market needed to present finished software products to potential buyers in a number of different locales is a justified concern in competitive settings from a marketing point of view, in particular when several regional markets are targeted simultaneously with localised versions of a software product, as "companies which are the first to market a new product capture a larger share of the profits" (AitSahlia 1995: 168). Hence, keeping short turnaround times and streamlining working cycles for translating graphic user interfaces and accompanying documentation is typically a top priority in software localisation projects (cf. Topping 2000).

3. Software localisation and translation

3.1. Outsourcing localisation to single language vendors and multiple language vendors

Multilingual localisation projects are in general very difficult and laborious to manage, so they are very often outsourced by software houses or vendors to external localisation and translation agencies. This is most likely to happen if

software houses do not have in-house staff with the technical expertise or language skills necessary to localise their own products, which is often the case in companies that are not multinational or very large corporations (Thibodeau 2000:

Under such circumstances a viable option can be to outsource software localisation services altogether, hiring external contractors such as the so-called single language vendors (or SLVs, i.e. specialised translation agencies or freelance professionals that provide localisation services into one language), or multiple language vendors (MLVs, namely agencies or companies that are able to offer localisation services into a number of languages). Whether they are in-house departments of large software manufacturers, hired freelancers or external agencies (both SLVs and MLVs), providers of software localisation services are expected to deploy effective tools, resources and strategies to reduce delivery turnaround times to a minimum, thus keeping the time-to-market as short as possible (Combe 2000: 97 and Thibodeau 2000: 145-146).

3.2. Localisation and translation in progress

For some marketing policies, releasing localised versions of the same piece of software in different areas of the world at the same time (see section 2.4. above) is so compelling that "to achieve simultaneous language releases, a company must develop localized versions in sync with original product develppment" (Thibodeau 2000: 136). Strategies of this kind are aimed at minimising the amount of time spent on the overall development of a new product, and are very common across a number of industrial sectors, not only in software development and localisation. AitSahlia *et al.* (1995), for example, explain the challenges involved in what they call "concurrent engineering", and question the widespread assumption that developing different parts of a product simultaneously is generally preferable to a linear and sequential approach.

Designing software packages, especially operating systems or suites of multiple programs, is a very complex and elaborate task that goes through a series of stages and repeated cycles (from initial design through to quality assurance tests, debugging, troubleshooting, etc.). Engineering problems or internal conflicts in the software may crop up at the end of the testing phases, or only when the product is tested on specific hardware platforms or software configurations to check its operability in the final stage of the production cycle, shortly before its release. When such difficulties arise, re-engineering is needed

to increase the robustness and portability of the software by removing inconsistencies or fixing bugs, which may call for changes in some of its functions, modifications to the graphic user interface (GUI) and, accordingly, to the support documentation.

During software localisation projects the teams of translators working into the target languages chosen for the localised versions are usually asked to start work when the original software is still under development (Esselink 2000: 111, 443). They may hence base their translations on strings in the source language of the graphic user interface or texts in the support documentation relating to a beta version, i.e. a provisional and partial development of the software that is still incomplete or needs to be checked and tested, and is thus subject to corrections if troubleshooting is needed after discovering internal bugs[3]. This approach to software development entails that the translation process for localisation sometimes takes place while the original version of the software is still being tested and modified, so that in some cases the two activities tend to overlap considerably. Also, the translation process may be particularly time-consuming for some of the languages involved in the localisation project (e.g. because they need more resizing or re-shaping of text boxes than others).

Even when a localisation project for multiple languages is at a fairly advanced stage and close to completion, the initial native version of the software product may undergo changes, needed because the client modifies certain functions or graphic elements (e.g. icons, glyphs, arrangement and strings of dialogue boxes, etc.), the engineers fix bugs, project managers revise the initial plans and the allocation of resources, or reorganise the task scheduling, etc. Modifications to the software in the original source language that occur 'in progress' clearly affect the content and information of the derived localised products, resulting in a sort of cascade effect.

In such cases, then, it is more than likely that changes will affect not only the strings and commands that are viewed on the interface and the screen shots, but also the descriptions and overall structure of accompanying documentation for each target language at different levels (e.g. printed user

---

[3] Some authors (e.g. MacCormack *et al.* 2001: 140) make a distinction between prototype and beta version, to mark the various steps that eventually lead to the development of a fully-fledged commercial software product. Although it is recognised that there are usually differences between the two (with beta versions being the result of more advanced stages in the process than prototypes), they are not considered here as they are not crucial to the main point of this discussion.

manuals, installation guidelines and tutorials, on-line context-sensitive help files, etc.). When substitutions or changes of some kind are introduced at any stage of a software localisation project (and in particular towards the end), there may be a need for translators to modify, edit, remove or add portions of the textual material that has already been translated. These sections may be entire blocks of text, isolated terminology items (e.g. in the header of a pull-down menu or the title of a dialogue box) or chunks of sentences and phrases.

I

3.3. Translation of menus, short-cut keys, hot keys and accelerators

These modifications typically affect short-cut keys (also called hot keys or accelerators), which are combinations of keyboard strokes involving letters and function keys (e.g. CTRL+SHIFT+Z) that enable users to interact with software applications, issue commands to operate them and invoke functions without the need to access a menu or use the mouse (cf. Esselink 2000: 70-73, 110, 122-123, Del Galdo & Nielsen 1996 and Hall & Hudson 1997). Short-cut keys are (commonly present in standard Windows-based programs, such as those with pull-down menus and dialogue boxes that can be accessed from the main working environment.

Users who do not wish to rely on using the mouse very often have the possibility of employing the keyboard to interact with the software and guide some of its operations and functions. By pressing short-cut keys on the keyboard, they typically issue the same commands as they would do with the mouse, but the keyboard-based procedure is usually quicker, and some users feel more comfortable with it, especially the more experienced ones. The letters that are found in hot keys are typically associated mnemonically with the description of the intended command (e.g. in the English version of popular Windows-based programs, 'CTRL+N' may stand for 'New document' or 'New file, 'CTRL+F' for 'Find', etc.), so as to facilitate the memorisation of keyboard-based hot keys and accelerators.

There should also be a link between the headings used for commands in the graphic user interface (e.g. on the items of pull-down menus that can be activated by clicking on a toolbar) and the corresponding combination of the short-cut key. As is often the case in the Windows-based environment, for instance, menu headings and strings show the associated letter underlined on the GUI, so that users are reminded at a glance of the relevant key they are

supposed to hit, if they prefer to issue a command by means of the keyboard rather than by moving the mouse pointer and clicking with it.

If the software is modified after testing, debugging and troubleshooting take place, there may be a need when translating the text strings of the GUI to change the short-cut keys used in the initial version. These changes should be carefully handled to ensure that no conflicts arise, as would be the case if one single short-cut key were associated with two different commands (cf. Esselink 2000: 70-73, 122-123). However, professional software localisation toolkits help translators by offering fully automatic or semi-automated cross-checks of the short-cut keys incorporated in every single application, identifying duplicated short-cut keys and overlapping combinations, and revealing inconsistencies and conflicts that can then be resolved by translators and localisers before the project is completed.

Furthermore, user-friendliness and usability of software applications are greatly enhanced if, for instance, short-cut keys associated with the basic operating functions are kept consistent across updated versions of the same program In a particular language. If, on the other hand, they change after an upgrade of the software package, users are unavoidably confused, and it would take some time and effort on their part to learn the new combinations of the hot keys and memorise which familiar functions and operations are associated with them.

3.4. Project management and leveraging material from past software localisation projects

For reasons of consistency and in order to maximise the time-efficient reuse of existing resources, software companies in general expect the translation of graphic user interfaces and all support documentation for newly released and upgraded versions of their products to be based on the multilingual documentation already available in the desired language(s) for similar products, and to be improved, adapted and updated accordingly. Clients require localisation providers to leverage the legacy material already available for the new localised version of their product to the largest possible extent, e.g. by using translation memory databases and producing translations according to validated glossaries and approved terminology lists.

Any successful approach to effectively exploiting the standardised resources available calls for task-oriented supervision and overall project management at all stages of software localisation, some of which have been outlined above. From the planning phases of localisation for a particular product, project managers maintain contacts and relations with clients, make sure that scheduling and budgeting agreements are feasible and adhered to, coordinate the employment of human professionals and allocate technical resources Esselink 2000: 427-428).

First of all, allocating the necessary resources in a software localisation project has to take budget restrictions and deadlines into account; however, it also encompasses other points, such as setting up teams of translators who are able to cover the desired target language(s), and of engineers who can deal with computing quality assurance checks, the provision of software and hardware platforms to test portability and robustness of localised programs, etc. One of **he** crucial tasks of project managers in multilingual software localisation projects is to ensure that the multiple versions of the product in all the required target languages are developed in parallel, with none of them lagging behind the time schedule, so that simultaneous release in a variety of locales is eventually possible without delays (in this respect, see sections 2.4. and 3.2. above).

4. Conclusion

4.1. New working practices for translators in software localisation projects

New practices are emerging as standard procedures to speed up and streamline the workflow of time-consuming translation jobs entailed by multilingual software localisation projects (cf. Thibodeau 2000: 136-141, 145-146). Job-sharing, for instance, means that teams of translators work connected to the same Local Area Network (LAN), accessing the same linguistic databases and resources, such as approved term-banks and validated translation memory archives updated in real time, in order to maximise the leverage from past similar localisation projects and optimise translations of relevant documentation by enhancing consistency as well. Similarly, multi-tasking is the approach adopted when teams of translators are divided up into smaller operative units, and each member carries out his or her part of the job from a different remote location. In this scenario each translator can liaise with his or her supervisor or project manager, circulate information to other colleagues involved in the project or address queries to the clients and software developers via the Internet, e.g. by e-mail.

These viable working conditions that rely heavily on technology and the telecommunications infrastructure can facilitate bringing together pools of professional translators for a variety of target languages (cf. O'Hagan 1996, Bowker 2002: 17-18 and O'Hagan & Ashworth 2002): as a matter of fact, software manufacturers and companies have the option of outsourcing localisation services and hiring single language vendors in the countries of the target languages needed, while maintaining some form of centralised co-ordination and supervision of the entire project. In such a situation, then, project managers can supervise the work of several localisation teams scattered around international locations from the headquarters of the software company: this strategy of centralised co-ordination and supervision may help to reduce translation costs, minimise turnaround times of translation teams, and maximise the efficiency of localisation processes Into multiple languages, taking into account the constraints imposed by the time-to-market factor (cf. Thibodeau 2000: 145-146 and Topping 2000: 115-116).

4.2. CAT tools are the key to coping with the time-to-market factor in localisation projects

Software localisation is at present a very challenging, potentially lucrative and profitable activity, which shows that translators have been greatly affected by the advent and enormous development of computing facilities and by the explosion of advanced telecommunications services. Professional translators play a key role in this innovative and highly complex commercial scenario, and are deeply involved in the workings of very dynamic processes during multilingual software localisation projects. Computer technology, telecommunications and IT provide translators with some of the solutions that are essential to success in an increasingly competitive global market, especially in software localisation.

By looking at the crucial role played by the time-to-market factor in the translation processes entailed by software localisation projects, this paper has discussed some of the main reasons why computer-assisted translation tools and efficient project management can prove successful in coping with it. Technological evolution is here to stay and is proceeding at an unprecedented pace: certain innovations, e.g. in computer-assisted translation, telecommunications and IT, deserve careful consideration, since today they offer invaluable support to translators, who can derive many real benefits from them, as shown by the cutting-edge business of software localisation.

Bibliography

AITSAHLIA (F.), JOHNSON (E.) & WILL (P.) 1995, "Is concurrent engineering always a sensible proposition?", in *IEEE Transactions on Engineering Management*, 42(2), pp. 166-170.

ALI (A.), KRAPFEL (R.) & LABAHN (D.) 1995, "Product innovativeness and entry strategy: impact on cycle time and break-even time", in *Journal of Product* Innovation Management, 12(1), pp. 54-69.

AUSTERMÜHL (Fr.) 2001, *Electronic tools for translators*, Manchester, St. Jerome.

BAYUS (B.L.) 1997, "Speed-to-market and new product performance trade-offs", in *Journal of Product Innovation Management*, 14(6), pp. 485-497.

BOWKER (L.) 2002, *Computer-aided translation technology: a practical introduction*, Ottawa, University of Ottawa Press.

BOWKER (L.) 2003, "Terminology tools for translators", in *Computers and translation: a translator's guide*, Harold Somers ed., Amsterdam/Philadelphia, John Benjamins, pp. 49-65.

COHEN (M.A.), ELIASHBERG (Jehoshua) & HO (Teck-Hua) 1996, "New product development: the performance and time-to-market tradeoff", in *Management Science*, 42(2), pp. 173-186.

COMBE (K.) 2000, "Localization at Hewlett-Packard's LaserJet Solutions Group", in *Translating into success: cutting-edge strategies for going multilingual in a global age*, Robert C. Sprung ed., Amsterdam/Philadelphia, John Benjamins, pp. 97-109.

DEL GALDO (E.) & NIELSEN (J.) eds., 1996, *International user interfaces*, New York, John Wiley & Sons.

ESSELINK (B.) 2000, *A practical guide to localization*, Amsterdam, John Benjamins.

ESSELINK (B.) 2003, "Localisation and translation", in Computers and translation: a translator's guide, Harold Somers ed., Amsterdam/Philadelphia, John Benjamins, pp. 67-86.

GORDON (I.) 1997, "The TM revolution - What does it really mean?", in *Proceedings of the conference Translating and the Computer 19,* London, Aslib.

HALL (P.A.V.) & HUDSON (Raymond) eds., 1997, *Software without frontiers: a multi-platform, multi-cultural, multi-nation approach,* New York, John Wiley & Sons.

HEYN (M.) 1998, "Translation memories: insights and prospects", in *Unity in diversity? Current trends in translation studies,* Lynne Bowker, Michael Cronin, Dorothy Kenny & Jennifer Pearson eds., Manchester, St. Jerome, pp. 123-136.

HUTCHINS (J.) 1998a, "Translation technology and the translator", in *Machine Translation Review,* 7, April 1998, pp. 7-14. Available on-line at the URL http://www.bcs.org.uk/slggroup/nalatran/nalamtr7/nalamt76.htm [accessed 20th August 2004].

HUTCHINS (J.) 1998b, "The origins of the translator's workstation", in *Machine Translation,* 13(4), pp. 287-307. Available on-line at the URL http://ourworld.compuserve.com/homepages/WJHutchins/MTJ-1998.pdf [accessed 20th August 2004].

KENNY (D.) 1999, "CAT tools in an academic environment: what are they good for?", in *Target,* 11(1), pp. 65-82.

LUONG (T.V.), LOK (J.S.H.), TAYLOR (D.J.) & DRISCOLL (K.) 1995, *Internationalization:developing software for global markets,* New York, John Wiley & Sons.

MACCORMACK (A.), VERGANTI (R.) & IANSITI (M.) 2001 ."Developing products on 'Internet time': the anatomy of a flexible product development process", in *Management Science,* 47(1), pp. 133-150.

MERKEL (M.) 1998, "Consistency and variation in technical translations. A study of translators' attitudes", in *Unity in diversity? Current trends in translation studies,* Lynne Bowker, Michael Cronin, Dorothy Kenny & Jennifer Pearson eds., Manchester, St. Jerome, pp. 137-149.

MORGAN (L.O.), MORGAN (R.M.) & MOORE (W.I.) 2001, "Quality and time-to-market trade-offs when there are multiple product generations", in *Manufacturing & Service Operations Management*, 3(2), pp. 89-104.

O'BRIEN (S.) 1998, "Practical experience of computer-aided translation tools in the software localization industry", in *Unity in diversity? Current trends in translation studies*, Lynne Bowker, Michael Cronin, Dorothy Kenny & Jennifer Pearson eds., Manchester, St. Jerome, pp. 115-122.

O'HAGAN (M.) 1996, *The coming industry of teletranslation: overcoming communication barriers through telecommunications*, Clevedon, Multilingual Matters.

O'HAGAN (M.) & ASHWORTH (D.) 2002, *Translation-mediated communication in a digital world: facing the challenges of globalization and localization*, Clevedon, Multilingual Matters.

SMITH (Pr.G.) 1999, "From experience: reaping benefit from speed to market", in *Journal of Product Innovation Management*, 16(3), pp. 222-230.

SOMERS (H.) 2003a, "The translator's workstation", in *Computers and translation: a translator's guide*, Harold Somers ed., Amsterdam/Philadelphia, John Benjamins, pp. 13-30.

SOMERS (H.) 2003b, "Translation memory systems", in *Computers and translation: a translator's guide*, Harold Somers ed., Amsterdam/Philadelphia, John Benjamins, pp. 31-47.

THIBODEAU (R.P.) 2000, "Making a global product at Mapinfo Corporation", in Tr*anslating into success: cutting-edge strategies for going multilingual in a global age*,Robert C. Sprung ed., Amsterdam/Philadelphia, John Benjamins, pp. 127-146.

TOPPING (S.) 2000, "Shortening the translation cycle at Eastman Kodak", in Tr*anslating into success: cutting-edge strategies for going multilingual in a global age*, Robert C. Sprung ed., Amsterdam/Philadelphia, John Benjamins, pp. 111-125.

UREN (E.), HOWARD (R.) & PERINOTTI (T.) 1993, *Software internationalization and localization*, New York, Van Nostrand Reinhold.