Localisation and Internationalisation

1.What is localisation?

The Localisation Industry Standards Association (LISA) defines the process of localisation (often abbreviated to "L10n") as:

> taking a product and making it linguistically and culturally appropriate to the target locale (country/region and language) where it will be used and sold (Esselink, 2000, p.3)

Thus, a locale is a combination of language and country or region.

Firstly, this definition recognises that more than one language may be used in a single region or country. For example, Belgium has three official languages: Flemish, French and German. Conversely, (variants of) a single language, such as Spanish, is often used in many different countries.

Secondly, it recognises that appropriateness extends beyond linguistic considerations.

As such it is not sufficient in all cases to provide different "language versions". Although people of different regions may officially speak the same language, there may be linguistic and non-linguistic differences between them that require specific treatment. For example, the Belgian French dialect (Walloon) differs from the French used in Canada, France or Senegal. Moreover, cultural symbols may differ. The French flag is not the flag of Belgium.

As a process, localisation is typically applied to software products. A relatively simple localisation project might involve the adaptation of a website. Larger projects may involve a range of materials that combine to create a single product, for example a software application (all language-specific features of the user interface, such as menu entries, tool tips, dialogue boxes, error messages etc), together with a combination of electronic and printed supporting documentation, such as online help, web-based materials and user manuals.

The range of file formats that may be involved in software localisation projects is extensive. Such formats include binary software files (.exe and .dll) "resource files" (.rc), web documents (HTML and XML), online help (WinHelp and HTML Help), office formats (such as Microsoft .doc, and .rtf) and desktop publishing (DTP) formats (such as FrameMaker and Quark).

What does localisation involve that is not part of traditional translation?

Localisation involves various activities and the consideration of features which might not be relevant to more traditional forms of translation.

Traditional translation projects typically involve a number of activities, in addition to the act of translating:

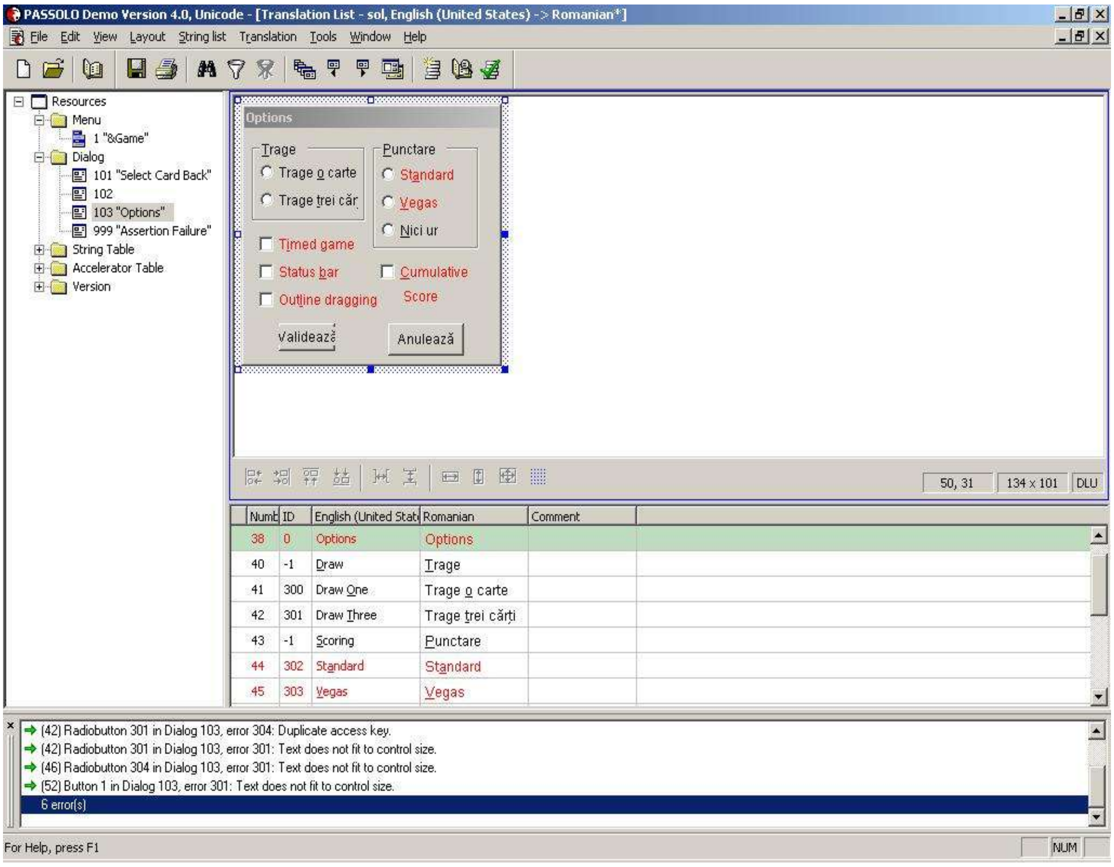
- Terminology research
- Editing
- Proofreading
- Page layout

Localisation tends to involve additional activities, such as:

- Multilingual project management
- Advice on translation strategy
- Conversion of file formats in which material is held
- Alignment and maintenance of translation memories
- Software and online help engineering and testing
- Multilingual product support

These additional activities are significant because of the increased range of skills they require. Often a localisation agency will concentrate on these specialist activities, and out-source the actual translation work to freelancers.

With regard to the technical problems involved in software localisation, specialist applications are available. These include Alchemy CATALYST and the PASS Software Localizer (PASSOLO). Typically, these combine the TM functions discussed in Lecture 5 with facilities for the extraction of translatable text from binary software files and warnings about some of the problems that will be mentioned below.



Numb	ID	English (United Stat	Romanian	Comment
38	0	Options	Options	
40	-1	Draw	Trage	
41	300	Draw One	Trage o carte	
42	301	Draw Three	Trage trei cărți	
43	-1	Scoring	Punctare	
44	302	Standard	Standard	
45	303	Vegas	Vegas	

```
(42) Radiobutton 301 in Dialog 103, error 304: Duplicate access key.
(42) Radiobutton 301 in Dialog 103, error 301: Text does not fit to control size.
(46) Radiobutton 304 in Dialog 103, error 301: Text does not fit to control size.
(52) Button 1 in Dialog 103, error 301: Text does not fit to control size.
6 error(s)
```

(Screenshot from PASSOLO showing a partially localised dialogue box and a series of warnings about problems (towards the bottom of the window))

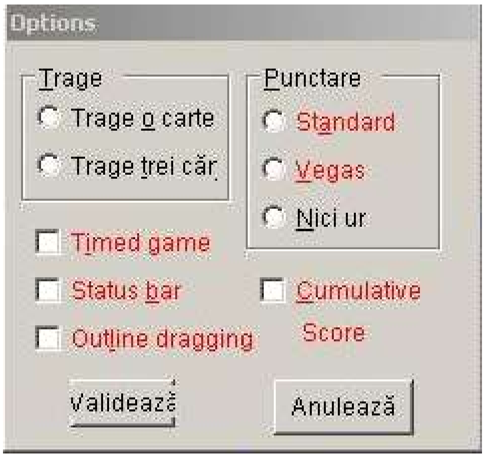
Some specific considerations

Having looked briefly at the range of activities and skills that are required in localisation, it is worth discussing some of the specific considerations in a software localisation project.

Text expansion

Thought needs to be given to the amount of space that translated text will occupy in a localised user interface (UI). Such text strings will appear in menu entries, window titles, dialogue boxes and as labels for buttons. There is little point in translating menu entries if they cannot be read, because there is insufficient space to display them!

In general, a translation will use more words than the original. Moreover, different languages have their own properties. The average German word is longer than the average English word. The equivalent word in Chinese is likely to use fewer characters and therefore require less space.



(example of a dialogue box which has been partially localised for a Romanian audience – notice that the text on the second left hand radio button (under “Trage”) does not fit the available space, neither does that on the left hand push button)

Web pages and applications may be designed to be flexible in the way space is allocated. This will allow areas to expand and contract depending on the amount of space required.

However, more often than not, the translator must work within confines that have already built into the material that she is translating.

Hot-key assignment

Another issue surrounds the assignment of hot-keys, which provide shortcuts to certain functions in many applications. These are often identified in the user interface by underlining the letter of the assigned key. In the example above, “t” has been assigned twice, thus creating ambiguity which will cause the localised program to fail.

Specialised software localisation tools, such as those mentioned above, will often warn the translator if the allowed amount of space has been exceeded or if conflicting hot-keys have been assigned (see the bottom panel of the large PASSOLO screenshot).

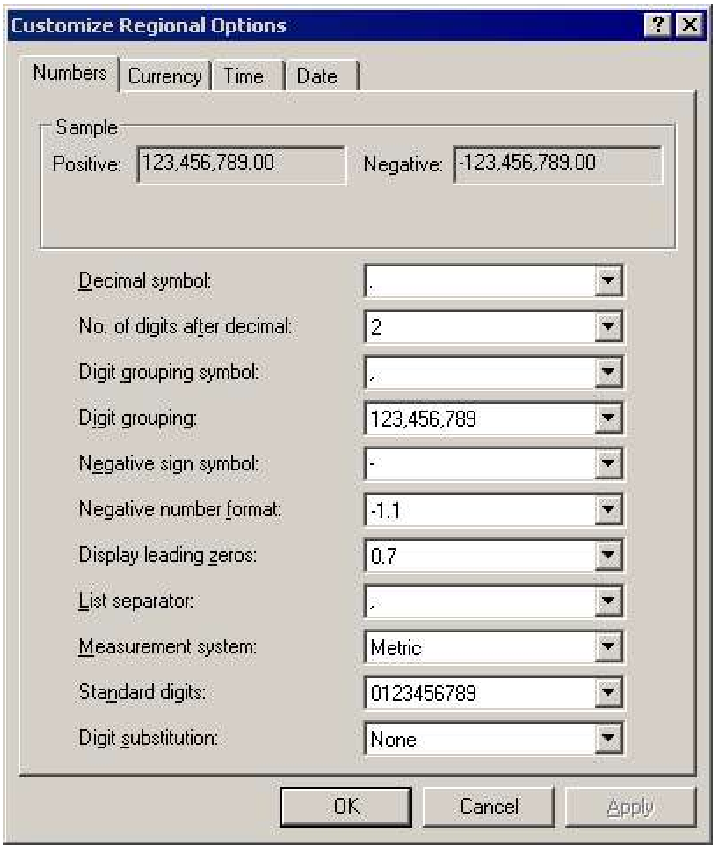
Product names

Some product names have unfortunate connotations in a target locale. In English, “Nova” might be an entirely appropriate name for a car. In Spanish, “*no va*” translates as “does not go.”

Dates, weights, and other numbers (and addresses)

Text is not the only aspect of products which can affect localisability. Hence the reference to culture in the LISA definition cited above.

Often information such as dates, weights and measures, currency values and address components are written and read differently in different parts of the world. Moreover, in some situations, formal constraints are imposed in order to ensure that only “legal” values are entered by users. For example, US zip codes consist solely of numbers, whereas UK post codes have both number and letters. A US-designed database, or even a web form, might be unable to accommodate UK data, because it accepts only numerical values in the zip code/post code field.



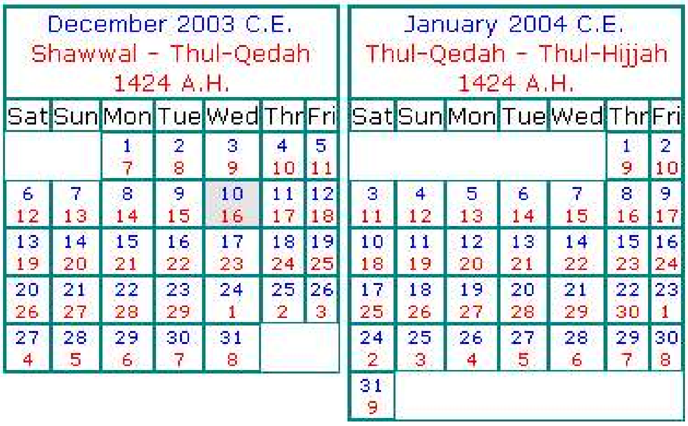
Sample
Positive: 123,456,789.00
Negative: -123,456,789.00

Decimal symbol: .
No. of digits after decimal: 2
Digit grouping symbol: ,
Digit grouping: 123,456,789
Negative sign symbol: -
Negative number format: -1.1
Display leading zeros: 0.7
List separator: ,
Measurement system: Metric
Standard digits: 0123456789
Digit substitution: None

Imperial measures may be used and understood in the UK, but they are not used in Russia. In addition to translating text into Russian, it might be necessary to convert values.

Dates are another area where care should be taken. While British English tends to describe a date as “10 December”, or “10th December”, Americans are more likely to say “December 10”.

The calendar itself is a function of culture and many calendars are still in use for various purposes. The Gregorian calendar most widely used in the West began on the traditional date of Christ's birth. Traditionally, Chinese and Japanese calendars have counted years since an Emperor’s accession to the throne. The Islamic Hijri calendar counts years in lunar months since (the Gregorian) 622 AD/CE, the year of the Hijrah, or migration, from Mecca to Medina.



December 2003 C.E.
Shawwal - Thul-Qedah
1424 A.H.

Sat	Sun	Mon	Tue	Wed	Thr	Fri
		1 7	2 8	3 9	4 10	5 11
6 12	7 13	8 14	9 15	10 16	11 17	12 18
13 19	14 20	15 21	16 22	17 23	18 24	19 25
20 26	21 27	22 28	23 29	24 1	25 2	26 3
27 4	28 5	29 6	30 7	31 8		

January 2004 C.E.
Thul-Qedah - Thul-Hijjah
1424 A.H.

Sat	Sun	Mon	Tue	Wed	Thr	Fri
					1 9	2 10
3 11	4 12	5 13	6 14	7 15	8 16	9 17
10 18	11 19	12 20	13 21	14 22	15 23	16 24
17 25	18 26	19 27	20 28	21 29	22 30	23 1
24 2	25 3	26 4	27 5	28 6	29 7	30 8
31 9						

(illustration taken from: http://www.islamicfinder.org/)

Even simple numbers can be presented differently, giving rise to confusion. Consider the German number “1.345”. This would be represented as “1,345” in English. Careless localisation could lead people to misinterpret this figure quite dramatically!

It could be argued that such non-linguistic aspects have long been present in translation. While this is true, it is worth noting the dramatic increase in the transfer of information around the world recently and also that the literary focus of earlier translation has shifted towards technical, financial and other commercial material.

Graphics

Graphics, such as those used in icons, sometimes have text embedded within them. As with text embedded within program code, this practice can hinder localisation because of problems of extraction.

Even purely pictorial icons must be considered, as they often have culture-specific connotations. How many people in England pick up their “post” from an American style “mail-box”, sitting on a post at the end of the garden? We may know what one looks like, from American TV and film (or “movies”!), but the image is not necessarily universal. The same could be said for a knife and fork used to indicate food. Chopsticks might be more lucid in some settings.

Finally, as we have seen in the opening remarks, flags do not make good indicators of language.

2.What is internationalisation?

The answer is given in another LISA definition:

> Internationalization is the process of generalizing a product so that it can handle multiple languages and cultural conventions without the need for re-design. Internationalization takes place at the level of program design and document development. (quoted in Esselink, 2000, p.2)

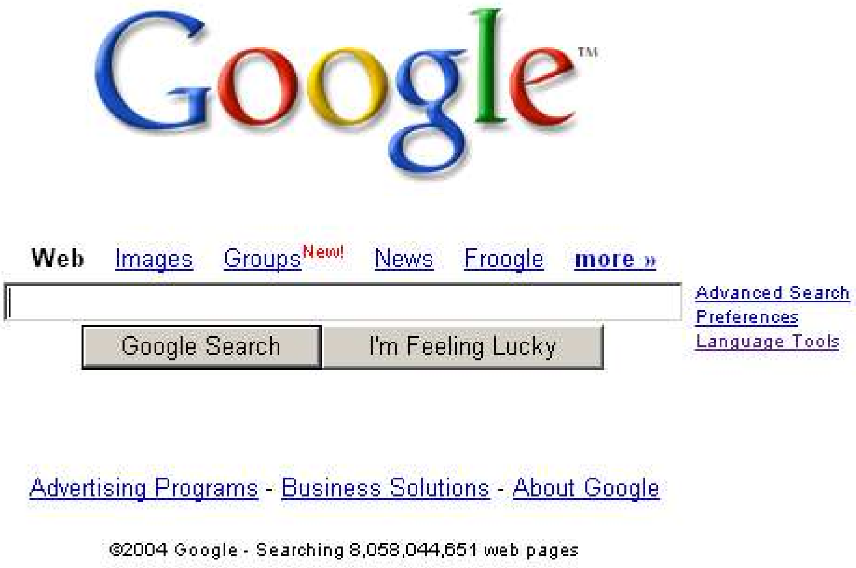
In other words, internationalisation (often abbreviated to “i18n”) is the creation of products in such a way as to facilitate localisation. We have touched on one or two specific issues of design above. Now we will turn to a more general discussion of these considerations.

Internationalisation and product design

Some materials are more readily localised that others. Good design, of both software and documentation, can facilitate the localisation process, just as a lack of planning can impede it. Good design is also likely to improve the product and reduce costs in other ways. For instance, it should simplify modifications or upgrades.

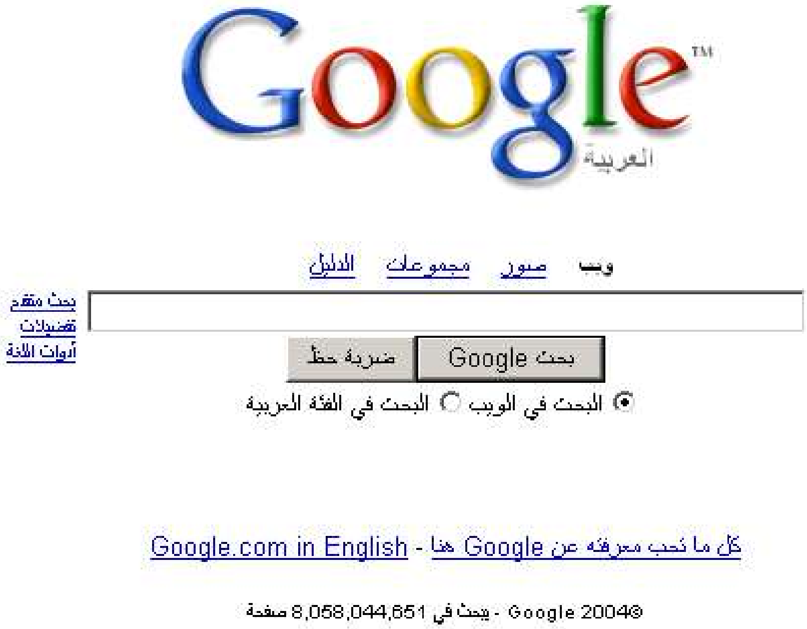
User Interface (UI) layout

Conventional design will prioritise the most significant buttons and menus by putting them where the user will see them first. If a language is read from left to right, as in the case of English, this means putting them on the left.



©2004 Google - Searching 8,058,044,651 web pages

If a language is read from right to left, as in the case of Arabic, this means putting them on the right.


Google™
العربية
وب
صور
مجموعات
الدليل
بحث متقدم
تفضيلات
أدوات اللغة
بحث Google
ضربة حظ
● البحث في الويب ○ البحث في الفئة العربية
Google.com in English - كل ما تحب معرفته عن Google هنا

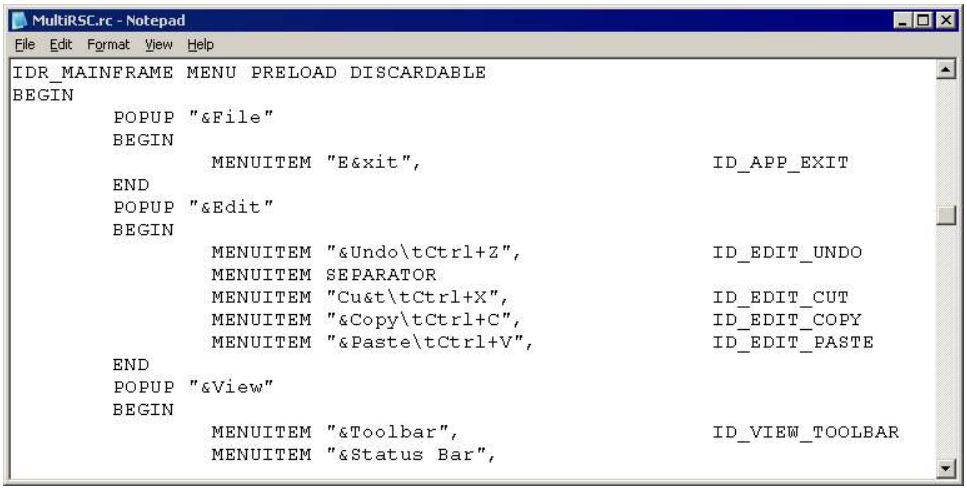
©2004 Google - يبحث في 8,058,044,651 صفحة

In addition to designing the user interface in such a way as to accommodate right-to-left languages, whilst maintaining a consistent look and feel, it might be important that supporting documentation be written so as to avoid references to "the button on the left" and similar layout-specific information – this will make the localisation process less error-prone.

Handling multilingual text

Separate the translatables

The first step to ensuring that software applications are easily localisable is to avoid embedding, or hard-coding, translatable text strings for the user interface (UI) within the program code.



```
IDR_MAINFRAME MENU PRELOAD DISCARDABLE
BEGIN
        POPUP "&File"
        BEGIN
                MENUITEM "E&xit",                       ID_APP_EXIT
        END
        POPUP "&Edit"
        BEGIN
                MENUITEM "&Undo\tCtrl+Z",               ID_EDIT_UNDO
                MENUITEM SEPARATOR
                MENUITEM "Cu&t\tCtrl+X",                ID_EDIT_CUT
                MENUITEM "&Copy\tCtrl+C",               ID_EDIT_COPY
                MENUITEM "&Paste\tCtrl+V",              ID_EDIT_PASTE
        END
        POPUP "&View"
        BEGIN
                MENUITEM "&Toolbar",                    ID_VIEW_TOOLBAR
                MENUITEM "&Status Bar",
```

(example of a resource file containing translatable text strings)

Instead, text should be stored in separate "resource files". In this way, locale-specific files can be accessed by the program, depending on the user's preferences, without requiring multiple versions of the program itself.

Use of specialised software localisation tools, such as those mentioned above, makes the process of translating software strings relatively straightforward.

Character encoding

In addition to the location of textual content, the character encoding used within resource files and web pages is also significant. If a multilingual character encoding, such as Unicode, is used, then languages into which the product is to be translated should easily be supported. It should be noted that, where appropriate, programs should be able to handle multilingual input text using a suitable encoding scheme as well as simply using it to store text for the user interface (UI).

Single sourcing and documentation design

Any documentation accompanying a product, such as a software application, can also be designed so as to help (or hinder) localisation.

Consideration of the process of localisation when creating documentation in the original language will help to ensure that the maximum benefit can be derived from

CAT tools. On the other hand, problems in the source language are likely to multiply through translation into other languages.

In recent years, “single sourcing” has been promoted as the best strategy for producing documentation. Single sourcing has been defined as:

> using a single document source to generate multiple types of document outputs; workflows for creating multiple outputs from a document or database. (Society for Technical Communication quoted in Joe D. Williams, “The Implications of Single Souring for Technical Communicators”, in *Technical Communication*, August 2003)

Thus, a single source text can be chopped up differently and used to output documents in a variety of formats.

Such an approach offers many benefits. Significantly, savings, like costs, may be multiplied when material is to be reproduced in several languages.

Single sourcing can save time, money and effort. In essence, it reduces duplication.

Firstly, fewer documents need to be written. Rather than writing some online help, a webpage and a user manual, chunks of the same underlying material can be reused to generate the various formats.

Secondly, documentation is easier to maintain. Once material is updated in the single source, modifications will be reflected in all formats which are subsequently output.

Thirdly, the benefits accruing form the removal of duplication in the source is multiplied each time it is translated into a new target language.

In order for benefits associated with single sourcing to be realised, a number of things need to be taken into account.

Perhaps most importantly, authors of material to be incorporated into a single sourcing approach to publishing might need to work with smaller units (grains) of text than they are used to. Efficient redeployment of grains of text to create different output formats relies on a high degree of flexibility. In essence, this means a high degree of granularity.

Acknowledgement

These notes draw on material written previously for the eCoLoRe project: http://ecolore.leeds.ac.uk/