LOCALISATION AND LINGUISTIC ANOMALIES

George R S Weir[1], Giorgos Lepouras[2]

[1] Department of Computer Science, University of Strathclyde, Glasgow G1 1XH, UK
[2] Department of Informatics, University of Athens, Athens 157 71, Greece

ABSTRACT

Interactive systems may seek to accommodate users whose first language is not English. Usually, this entails a focus on translation and related features of localisation. While such motivation is worthy, the results are often less than ideal. In raising awareness of the shortcomings of localisation, we hope to improve the prospects for successful second-language support. To this end, the present paper describes three varieties of linguistic irregularity that we have encountered in localised systems and suggests that these anomalies are direct results of localisation. This underlines the need for better end-user guidance in managing local language resources and supports our view that complementary local resources may hold the key to second language user support.

1 INTRODUCTION

In the quest for wider computer access, many authors have stressed the need to accommodate users whose first language is not English. This is often expressed as recommendations for internationalisation while others focus on the desirability of software localisation (e.g., Uren et al, 1993, O'Donnell 1994, Belge, 1995). Internationalisation and localisation can be understood as inter-related aspects of 'globalisation', or the move toward provision of globally accessible interactive systems. Internationalisation is the process of developing applications that can easily be converted to operate in different cultural or linguistic environments (Russo & Boor, 1996). The cultural environment extends beyond the local language to aspects such as beliefs, customs and ethics of a society. The internationalisation process ensures that culture-specific characteristics of an application are isolated, thereby enabling easy localisation.

Localisation is the process of converting applications to operate in a specific cultural environment. Occasionally, software is localised without first being internationalised, though, if internationalisation has been employed during the development of the application, localisation time can be reduced.

The localisation process reasonably consists of two main stages. The first step is translation of language resources to reflect the local language. In this stage, all language resources are translated to the local language, including features such as menus, commands, help texts, etc. Occasionally, the translation process employs a local software company or a company with expertise in translation. However, the outcome of the translation process is not always as anticipated (Lepouras & Weir, 1999).

The second step in localisation adjusts software to local cultural habits. Here, the application is adapted to reflect local customs. If the application is already internationalised, this stage may be unnecessary, since changes, such as special sorting algorithms, may have already been met during the development stage. For symbolic features such as the currency or the comma delimiter, the application may simply inherit configuration from the operating system.

While generally endorsing the importance of such linguistic and cultural accommodations, our purpose in this paper is to highlight several concerns in the use of localisation. Through problem illustration in the context of Windows-based applications, we stress the need to understand the technical mechanisms that afford software localisation. Without adequate grasp of the operational alternatives and their possible impact, there is serious risk that localisation may lead to linguistic anomaly and sub-optimal user interaction. We conclude with a set of recommendations that aim to minimise the threat of anomalies when developing or working within a localised setting.

2 LINGUISTIC ANOMALIES

The illustrations below are drawn from the context of Microsoft Windows™ and its applications. In this setting, several localisation features are available and are in common use wherever English is not the native user language. Such features vary from local font support within the operating system through localised (extensively translated) applications. This fact, coupled with the prevalence of the Microsoft environment, affords a credible setting in which

to reveal the occurrence of linguistic anomalies. We have identified three varieties of anomaly: instances arising within mixed linguistic environments, terminological problems, and peculiarities resulting from extreme localisation.

2.1 Mixed environments

Perhaps the most apparent problem with localised applications arises in mixed environments. International users rarely enjoy a working environment that is fully localised. An environment where every application including the operating system speaks the user's native language is uncommon outside the English-speaking world. More often, the user's environment comprises a mixture of English and local language. This mixed language scenario may take two forms: between applications or within the same application. In either case, the dual language requirement impairs the uniformity and consistency of the user's working environment. This contravenes an explicit objective of most windowing environments (e.g., see Microsoft, 1992, Apple, 1992, Sun, 1999) and users confronted with such inconsistency face an additional burden of comprehension.

2.1.1 Mixed language between applications

In a mixed environment of localised and non-localised applications, the user has to learn both the original and the local language terminology to be able to interact with the applications installed. In such a context the putative advantage of using localised applications may become an extra overhead for the user. Imagine a user faced with a localised word-processing application and non-localised drawing application. The inevitable overlap of functionality requires that the user manage two contexts of interaction. In this case, two different applications share common menus, commands and underlying functions yet the user is faced with two sets of descriptors.

2.1.2 Mixed language in an application

Recently, another type of linguistic anomaly has emerged. This is characterised by the appearance of mixed language (English and local) within a single application. In such cases, the application maintains some resources in English and other resources in a local language. The result is often a peculiar mix of English and local language, with no obvious rationale for the mix. This is evident in **Figure 1** wherein the Help for MS-Word shows its first tab with a Greek label (Contents), while the remaining labels, and all other displayed text, appears in English.

Figure 1: Mixed Greek and English in a single application.

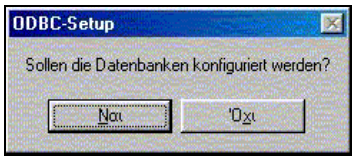


Figure 2: Mixed English, German and Greek in a single application.

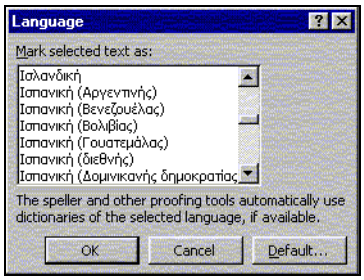
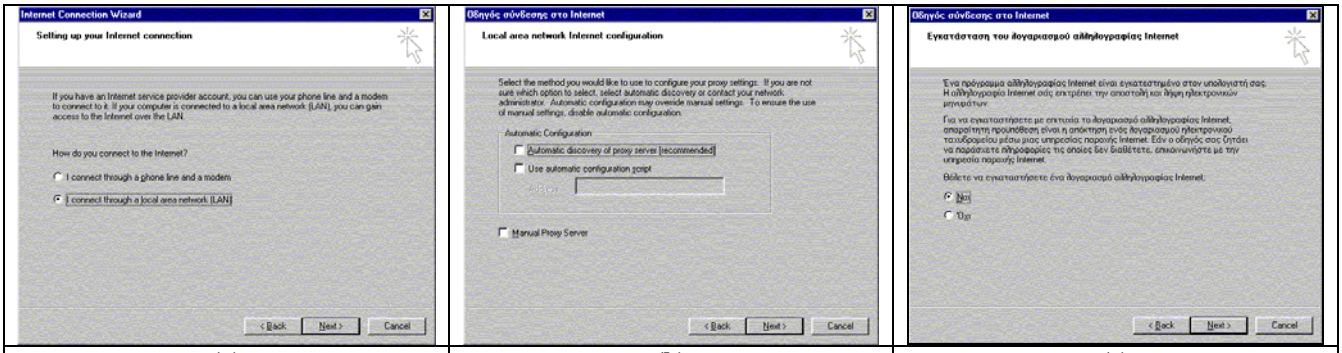


Figure 3: Further Mixed English and Greek in a single application.

A second illustration of mixed language proves that such linguistic anomaly is not exclusive to English and the local language. **Figure 2** shows the setup procedure of an application developed by a German software company. The title of the dialog box is in English, the text of the dialogue is in German and the button labels are in Greek!

This section's penultimate example comes from a non-localised word-processor. Figure 3 illustrates a dialogue box with a choice of dictionaries for use in a spell checker. Ironically, all dialogue text is in English, except for the choice of dictionary language. All of the languages appearing in this list box are named and listed alphabetically in Greek.

Our final example of intra-application language mixing is even more bizarre. **Figure 4** shows an initial screen (a) entirely in English. For no obvious reason, the next screen in the interaction (b) replaces the English title with a Greek language equivalent.



(a) **(b)** **(c)**

Figure 4: Mixed language within a single application
(Greek and English within Microsoft's Internet Connection Wizard)

Pressing the *Next* button in screen (a) takes us to screen (b). The linguistic strangeness continues at the third step in the dialogue. Pressing *Next* from screen (b) takes us to a third (more localised) display in which only the button labels remain in English (screen c).

With the possibility of such, apparently, arbitrary mixed language dialogues there is good reason to expect user confusion. One must take care however, to avoid the assumption that increased localisation will minimise user difficulties. There is strong evidence to suggest that full localisation without co-ordination between software houses leads to inconsistency across applications. Examples of such terminological linguistic anomalies are described below.

2.2 Terminological issues

Language problems are often caused by terminology. Whenever English language software is translated to a local language decisions are taken on mapping from English terms to local terms. Inevitably, some measure of arbitrariness is attached to this procedure. In consequence, some aspects of localised software may appear stranger to the local audience than the English (foreign language) original. This goes some way toward explaining why many users when faced with a choice between a localised (fully translated) application and an English-language original, express a preference for the latter (Lepouras & Weir, 2000).

Inconsistent localisation is evident between applications. For instance, localised Greek word processors show a measure of arbitrariness in choice of Greek terms. In a Greek version of Microsoft Word the term 'header' was replaced with 'κεφαλίδα' (derived from the Greek root for 'head'). A Greek version of Lotus Ami Pro replaced 'header' with 'υπέρτιτλος' (meaning 'supertitle'). Finally, a Greek version of WordPerfect has 'header' replaced with 'τίτλος σελίδας' (literally, 'page title'). A further selection of such terminological variety is given in Table 1 (cf. Lepouras & Weir, 1999).

English Term	Word 6.0	Lotus Ami Pro	WordPerfect
footer	υποσέλιδο	υπότιτλος	υποσέλιδο
bullet	κουκκίδα	σύμβολο	σφαίρα
undo	αναίρεση	ακύρωση	ακύρωση
view	προβολή	όψη	θέα

Table 1.: Example variations in localised terminology

Anomalies in terminology take forms other than variations in vocabulary. There are instances of neologism, transliteration (e.g., 'zoom' becomes 'ζουμ', 'style' becomes 'στυλ') and retention of original terms (e.g., 'Small Caps', 'textart' and 'kerning' are used verbatim in the localised versions of Greek word processors).

2.3 Extreme localisation

Our final category of linguistic anomaly derives from applications that are 'over-localised'. A classic example arises where a document was originally written in a localised word processor, and subsequently opened for editing in the original version of the same application. In this instance (**Figure 5**), the localised version had saved some style information (the paragraph numbering scheme) in Greek. As a result, when transferred to the original (English)

version of the word processor, errors were reported of an '*unknown switch argument*'. Viewing the underlying 'Field Codes' revealed that the Greek version had saved the numbering style as 'Αραβικά' instead of 'Arabic', yet the corresponding code for page numbering (i.e., 'PAGE') was not also translated from English.

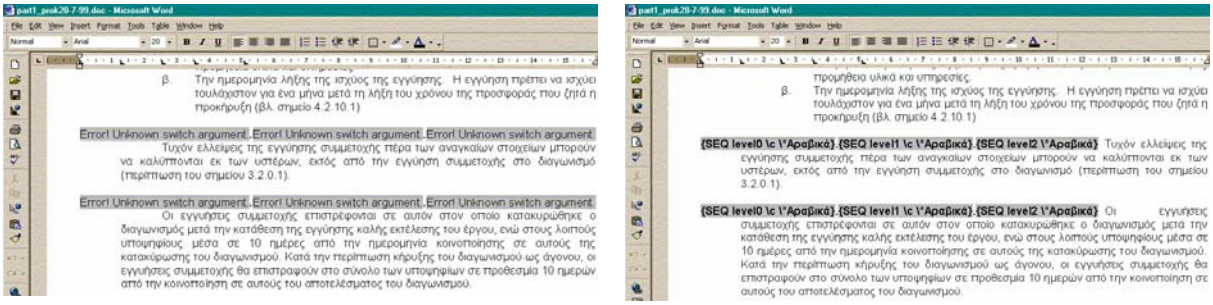


Figure 5: Extreme localisation of field codes

2.3.1 Criteria for localisation

There are varying degrees of localisation. Yet there are no obvious criteria for guiding the appropriate level of localisation. Evidence from our study of localised word processors shows that original menu shortcuts (such as Ctrl-C for 'Copy') are consistently retained rather than changed to accord with localised menu commands. The assumptions underlying this decision are obscure but may presume that retaining shortcut consistency across localised versions is beneficial to local users. Even this strategy can lead to anomalous results. Shortcut keys are often mnemonics for the English command names, e.g., Ctrl-N for 'New, Ctrl-O for 'Open' and Ctrl-S for 'Save' and Ctrl-P for 'Print'. When mapped to a localised Greek version of the 'File' menu, these mnemonics are inappropriate yet this set of shortcut keys from the English context are retained. This contrasts with the other set of application shortcut keys. Alt-F (mnemonic in English for 'File') invokes the File menu in the English version yet Alt-A is the equivalent Greek shortcut (mnemonic for 'Αρχείο'). In other words, one set of shortcuts is not localised (remains largely as English mnemonics) while another set is converted to localised mnemonics. Such examples illustrate a fundamental tension within localisation efforts, viz., the need to change interface characteristics whilst attempting to maintain consistency.

3 PROBLEM CAUSES

Inevitably, one must wonder about the causes for many of the linguistic anomalies described above. We must stress that no steps were taken to create artificially any of the reported examples. Our feeling is that many of the more peculiar anomalies result from vagaries of the localisation process. For instance, localised applications may invoke some non-localised language resources from the operating system. Similarly, non-localised applications may invoke localised language resources from the operating system. In either case, the result is likely to be linguistically anomalous. Extreme localisation, such as translating application codes and other invisible elements seldom accessed by users, are a potential risk. Reasonably, any term with operational effect (such as a parameter) should work in each language.

Some anomalies arise from conflicts between different versions of the same application, as in the transition from localised to non-localised word processor. Installed language resources may be compromised following installation of unrelated applications. Shared libraries are common in a Windows environment and not always evident to the end-user. In cases where users are asked to confirm the replacement of shared files there is an implied risk in either response and no information on the likely impact on common resources. The shared library approach is fundamental to the Windows architecture. For this reason, such linguistic anomalies persist through new releases of the operating system. **Figure 6** shows a now familiar syndrome in the context of Windows 2000.

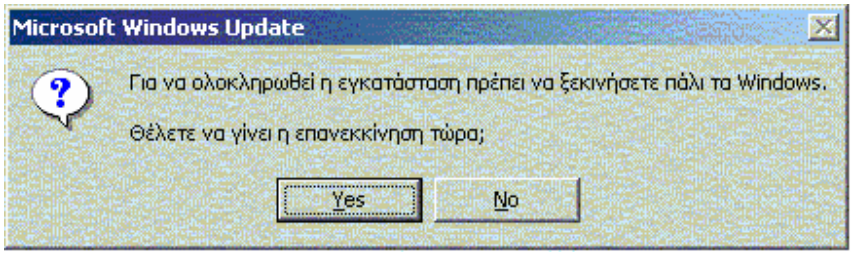


Figure 6: Linguistic anomaly in Windows 2000

4 CONCLUSIONS

Attempts to accommodate non-native speakers of English by translating or localising computer applications often give rise to unexpected difficulties. We have characterised three varieties of linguistic anomaly that may arise in such contexts. Although many of our examples are drawn from the Greek context, we are confident of similar experiences in other localised language communities. Notably, Pemberton (Pemberton, 2000) expresses similar concerns about localisation in a Dutch setting.

Many of the reported problems arise directly from localisation efforts, for instance, through conflicts in available language resources. Such anomalies can only occur because the language specific resources have been isolated from other functions of the application. We have focussed our descriptions on the user perspective. Evidently, some of the recounted anomalies arise through user efforts to deploy localisation resources within an environment that is often less than transparent. This suggests lessons both for end-users and for software designers.

End users who seek to deploy localisation facilities must be conscious of the coarse level of control over language resources. Additionally, users must appreciate how easily they may disturb the balance of localised resources within the operating software. The lesson to software designers is to facilitate easier and less tenuous language resource management. Responsibility falls on the designer to protect the end-user from anomalous effects. Ultimately, the range of potential difficulties, combined with the opacity of any applicable criteria is further reason for considering alternatives to localisation, e.g., through complementary support in the user's native language (cf. Weir et al, 1996, Lepouras & Weir, 1999, Lepouras, 2000).

5 BIBLIOGRAPHY

Apple, Macintosh Human Interface Guidelines, Addison-Wesley, Mass., 1992

Belge M., 'The Next Step in Software Internationalization', Interactions, ACM, 1995, Vol. 2. No 1, pp. 21-25.

del Gardo E., 'Internationalization and Translation: Some Guidelines for the Design on Human Computer Interfaces', in J. Nielsen (Ed.) Designing User Interfaces for International Use. Elsevier, New York, 1990, 1-10.

Lepouras G., User-Computer Interaction: The methodology of supplementary support in the service of different cultural communities, PhD Thesis (in Greek), Department of Informatics, University of Athens, 2000.

Lepouras G. & Weir G. R. S., 'Its not Greek to me: a cross language study of three word processors', SIGCHI Bulletin, ACM, 1999, Vol. 31 No. 2, pp. 17-24.

Lepouras G & Weir G. R. S., 'Mind your language: A study of language preference in Greek users', paper submitted to International Journal of Human-Computer Studies.

Microsoft, The Windows Interface: An Application Design Guide, Microsoft Press, Washington, 1992.

O'Donnell S. M., Programming for the World: A guide to Internationalization, PTR Prentice Hall, 1994.

Pemberton S., 'Reflections: so much for WYSIWYG', Interactions, ACM, 2000, Vol. 7, No. 5, pp. 60-61.

Russon, P. & Boor, S., 'How fluent is your interface? Designing for international users', Proceedings of InterCHI'93, ACM Press, pp. 342-347, 1993.

Sun, Java look and feel design guidelines, Sun Microsystems, California, 1999.

Uren E, Howard R, Preinotti T., Software Internationalization and Localization: An Introduction. Van Nostrand Reinhold, 1993.

Weir G.R.S., Lepouras G. & Sakellaridis U., 'Second-Language Help for Windows Applications', In M.A. Sasse, R.J. Cunningham, and R.L. Winder (Eds.), People and Computers XI, Proceedings of HCI '96 Conference, Springer, 1996, 129-138.