Translating software: what it means and what it costs for small cultures and large cultures.

Dai Griffiths
Ca'n Deià, Deià, Mallorca, Balears, SPAIN
Phone: (+34) 71 639 252
E-mail: dmidgr9@es.uib.ps

Stephen Heppell (Anglia Polytechnic University)
ULTRALAB,, Anglia Polytechnic University, Sawyers Hall Lane Brentwood, CM15 9BT, UK.
Phone: (+44) 1277 264 504 Fax: (+44) 1277 211 363
Phone: (+44) 277 264 504 Fax: (+44) 277 211 363
E-mail: heppell@applelink.apple.com

Richard Millwood (Anglia Polytechnic University)
ULTRALAB,, Anglia Polytechnic University, Sawyers Hall Lane Brentwood, CM15 9BT, UK.
Phone: (+44) 1277 264 504 Fax: (+44) 1277 211 363
Phone: (+44) 277 264 504 Fax: (+44) 277 211 363
E-mail: rmillwood@uk.ac.anglia.v-e

Greta Mladenova (Technical University Sofia)
ULTRALAB,, Anglia Polytechnic University, Sawyers Hall Lane Brentwood, CM15 9BT, UK.
Phone: (+44) 1277 264 504 Fax: (+44) 1277 211 363
E-mail: gmladenova@uk.ac.anglia.v-e

Abstract

In this paper the authors report as a case study their experience of adapting a set of software for other languages and cultures, drawing attention to the potential pitfalls and sharing what was learnt. This experience was based on a project to translate the "Work Rooms" software for young learners into Bulgarian and Catalan. It is also hoped to broaden the debate on CAL, stimulating consideration of multicultural and international issues.

While the questions raised by this particular adaptation of software are relevant to all those working with CAL, they have particular importance for software authors, publishers, and teachers of linguistic minorities.

Introduction

In this paper the question of adaptation of software for use in other cultures and languages is examined in terms of its benefits and in terms of the experience of the authors. Attempts to systematically organise software adaptation in this field appear to be few in number, indeed Van den Brande [1] reports that from the point of view of the European Community DELTA research and development program that:

"problems of interoperability, copyright, reusability of learning materials within different cultures are yet unsolved."

One reason for this may be that only recently has innovation in information technology made it straightforward to mix alphabets and languages in a single user interface with the richness demanded by educational software design for both the author and of course for the learner. Nevertheless work has been done, notably by Cox and Bosler [2] in the training of educational software authors to adapt English software for German. In Cox and Bosler's paper, it is made clear that despite differences in curriculum and culture little of the software was re-designed. This may be partly due to flexibility in use of the software and partly due to the way in which software still challenges traditional practice in any culture. Certainly little attention has been paid in the work reported here to developing some kind of multicultural pedagogy - it is hoped that the flexibility and innovation of the Work Rooms software is a catalyst for pedagogic development wherever it is used and that the teachers and parents who observe its use help its authors to gain insights into the differences and commonalties in use.

The Work Rooms software

The Work Rooms software is designed to provide a range of tools for young learners for painting, writing, simple database work and mathematical problem solving. The activities are mostly open ended and present opportunities for problem solving, exploration and preparation of tasks by the teacher or parent. Each activity is represented by a room in a house (see Fig. 1).

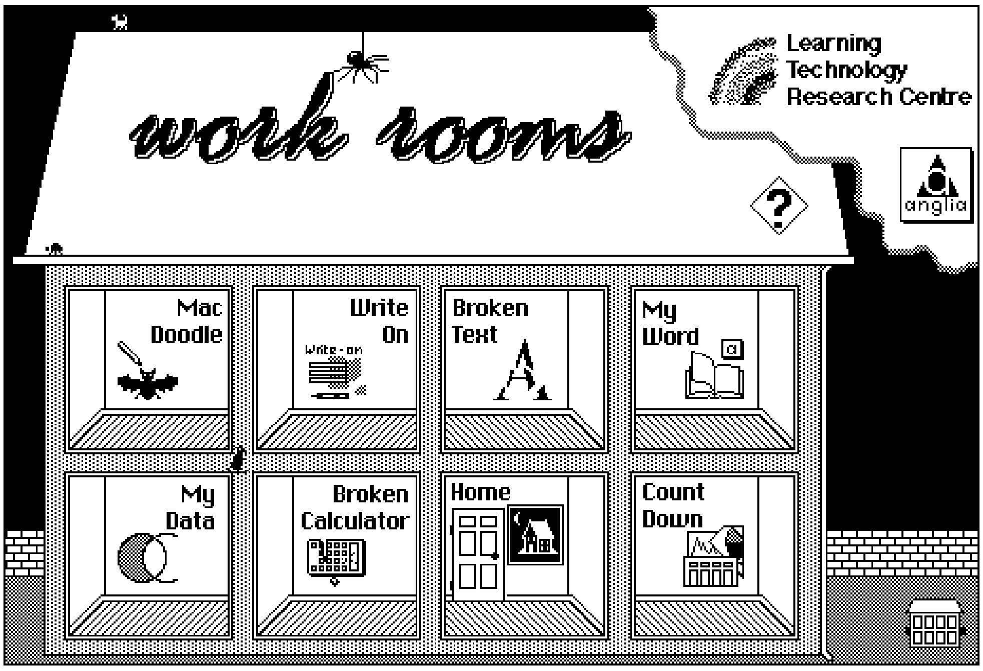


Figure 1 : The Work Rooms main screen

The software was designed in HyperCard using XCMDs (compiled extensions to the HyperTalk programming language) to deliver a number of features including context sensitive balloon help, tool palettes and sound. The software has been distributed in large numbers world-wide in English and has stimulated delighted reactions from children, parents and teachers.

The software has been translated into Catalan and Bulgarian (see Fig. 2), and work is in progress to translate into Danish, Norwegian, Russian and Welsh.

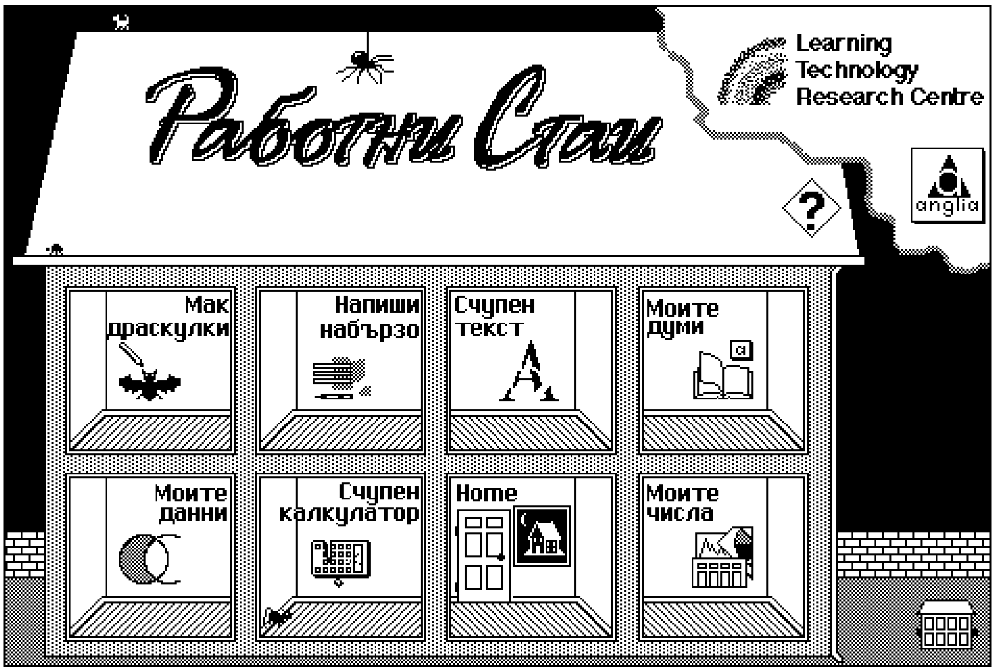


Figure 2 : The Work Rooms Bulgarian version

Why adapt software?

Commercial trade, television and tourism are often considered as threats to minority cultures, and now monolingual computers and their software join the list. The impact of CAI developed in one culture but then utilised in a multiracial culture is discussed by DeVillar and Faltis [3] in the context of US classrooms. DeVillar and Faltis see software which encourages communication, integration and co-operation as the solution. But this need not discourage us from adapting other styles of software for minority cultures, and at the same time agreeing with their conclusions. The adaptation of software for other cultures brings benefits in three areas: social, educational and financial.

Social benefits

Empowerment of individuals
It is pleasure to work with people who can interact with a computer in their own language for the first time, and see how relatively quickly they progress, and how much they enjoy it. It isn't known what the potential skills of these people are until they are given this opportunity, and they may have a significant contribution to make in an area where the computer can be of use.

Many people identify strongly with the culture in which they were brought up. If their language is not accepted into the magic circle of technology they may also feel personally rejected and second rate, and they may never summon up the confidence to approach IT. Alternatively they may have to split their linguistic life into discrete areas, using English for technological purposes and their own language elsewhere. This only serves to marginalise their language still further, and can also create problems by disturbing speakers' image of their personal identity and disrupting the ties in communities.

Support for smaller cultures.
The impact of information technology on the world's less powerful cultures has generally been debilitating. They do not have the financial resources to create programs in their own languages, which are consequently excluded from this linguistic domain. This has negative consequences, both for the culture and for the individuals speakers concerned. The exclusion of a language from advanced technology associates it with the past, and creates the unfounded impression that the language is intrinsically unable to cope with the demands of the modern world. It also makes it difficult to evolve adequate technical terminology, which reinforces the archaic image. Such a language frequently becomes marginalised and may be pushed out of existence altogether.

It is our position that information technology professionals should be aware of the implications which the technology they use has for less powerful cultures, and that they should try to reduce the damage which is caused. Using lesser-used languages in information technology gives those languages access to high status social functions, builds self esteem for their speakers and helps them both to develop their ability to deal with technical problems in their own languages and to learn more effectively.

Cultural diversity
The wide range of cultures in the world should be preserved, not only because of they may be "rich cultures" but also because it is important to maintain cultural diversity. A wider variety of cultures carries with it a wider variety of perspectives, potential insights and solutions to the world's problems.

The impact of information technology on cultural diversity has not, however, all been negative. The development of low cost desk-top publishing gave a new lease of life to publishing in a number of lesser-used languages: the increasing importance of software authoring tools for non-professional programmers is about to do the same for the creation of software.

Educational benefits

English as a foreign language
It is clear that a computer which has an English operating system and applications is perceived differently by first language English speakers and by users whose knowledge of the language ranges from bilingual to minimal. Exactly how people react to using a foreign language interface, however, is a complex matter. It cannot be assumed that people always want translations from English into their own language. Some non-native English speakers prefer to use English language software because it helps improve their command of a language which for good or ill is an essential key to success in the world of computing. There are, moreover, substantial practical advantages to being part of an international family of users.

Multilingual classrooms and homes
It is evident, particularly in urban centres, that multiple languages are used in both classrooms and even in homes. Software where learners can choose the language in which they work will enable study of other languages and adaptation to the range of learners needs in one classroom.

Human computer interface
On the other hand, a foreign language can add another layer of impenetrability to the human-computer interface. All except the most linguistically adept second language users have to devote much of their attention to interpreting the text on the machine, and have less opportunity to consider the choices presented on screen or comprehend the activities in which they are participating. New users need the least barrier to their understanding and the localisation of interfaces is an essential part of this. The localisation of interfaces and documentation also helps teachers and technicians solve the inevitable technical problems, so that computers remain available for use.

In order to overcome the lack of confidence many inexperienced computer users feel, authors attempt to make applications as friendly as possible, and full of positive reinforcement for the user. Unfortunately this may result in negative reinforcement for some users - cheerful colloquial messages in text or sound can add yet another level of impenetrability to the struggling second language user.

Effect of language on participation
We have observed learners whose English is weak using software written in English, and we have come to the unsurprising conclusion that foreign language interfaces tend to create a passive relationship with the machine. Users see such programs as belonging to another culture, and while this may make the computer even more fascinating to observe, it discourages learners from participating in activities. When speakers see their own language on the screen they engage with the activity in a different way, but the implications of this for the effectiveness of learning depend on the nature of the activity supported by the software. For instance trials with small numbers of children using the Work Rooms software in Bulgaria showed that when the software is in English they like to paint and draw, but not to write, whereas when the software is in translation they are happy to write creatively and investigate the whole software. Informal trials in the University of the Balearic Islands computer laboratory for schools showed similar results, and we are convinced that working in their own language significantly enhances learning for younger users.

One colleague, who teaches applications of computer' applications in the Technical University of Sofia to students in higher education, reports that even at this advanced level, where English is often mastered to a good standard, progress is more rapid with localised versions of software.

Financial benefits

Costs and returns
The disproportionate number of CAL applications written in English is a function of the preponderance of the English speaking market, which is undoubtedly the biggest in the world. The production of CAL software is an expensive exercise, however, and despite the size of the English speaking market, many projects never come to fruition because of lack of funding for educational institutions, or poor return on investment for potential investors. Adaptation of software for use in other cultures can dramatically change this situation even if we only consider major European languages. Spanish in particular is important because of its presence in the US market, and in California many schools consider bilingual Spanish/English software a high priority. It would not be unrealistic to imagine that a product produced in German, French and Spanish would have twice the market of a product produced only in English, and as the costs involved in adapting the software are relatively small, this can make the difference between success and failure for many commercial projects. Similarly for an educational institution looking to fund a project, if the costs of development and production can be shared with institutions in other countries many funding problems can be solved at a stroke.

Visibility and feedback
Much educational software is developed in research centres for the purpose of greater understanding of "what works" and design development. Adaptation to a range of languages can help to factor out some of the specific cultural issues which may cloud our understanding of learning with computers. More directly useful, good work is better disseminated and feedback from end-users is increased and enriched.

What are the problems?

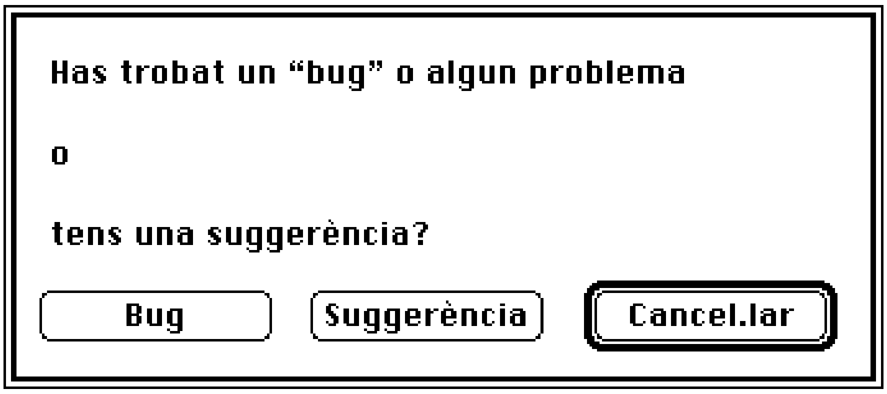
Language issues

English technical words
One problem for translators is deciding how far to go along with the borrowing of technical English in the target language, and how far they should actively create an independent technical vocabulary.

There are two main problems here.

Adoption of neologisms
Firstly, languages vary in the way in which they adopt neologisms. The major world languages are served by the International Organisation for Standardisation, founded in 1946, which among other activities aims to standardise equivalent technical terms in different languages. Many major languages also have an official body responsible for giving the official seal of approval to neologisms, such as the Academie Francaise or the Real Academia Español and the translator can simply consult their publications and find out what is permitted. The situation for lesser-used languages varies enormously, and some have no provision for creating new technical vocabularies, while at the other extreme the Catalan autonomous government has created the Institut d'Estudis Catalans, a body charged with approving neologisms, and TermCat, an extensive terminological data base which can be consulted remotely through a modem link.

Even when translators have found an officially sanctioned word, their troubles are not yet over. There may be other translations which are more widely understood and less academic, or IT professionals whose first language is not English may find that their native tongue looks strange on the computer screen, and may not easily accept variations from the accepted English vocabulary for the common terms in computer science. For example in the Work Rooms we had to translate "bug", which has an official translation in Catalan: "brossa". This is not a word in common usage however, and neither it nor the English original will be understood by the children, so it might be better to substitute "error" for "bug" (see Fig. 3). This unfortunately destroys the visual pun of a bug icon on the button, so we decided to use the English "bug" and include the translation of "error".


Has trobat un "bug" o algun problema

o

tens una suggerència?

Figure 3 : The translation of "bug" necessitated an additional translation

In other words translators have to determine what importance to give to the creation of a new vocabulary, and if the register of the new terminology is what is required. They have to decide if it is more important to be understood clearly by the few, or to participate in the creation of a new vocabulary which may seem forced and academic to some users, but which holds the promise of empowering more users in the future. This is not a problem confined to computing, and minority language speakers may have a similar experience when they watch a film in their own language. We have heard some Catalan speakers comment that while it is perfectly natural for Rambo or Tom and Jerry to speak Spanish, it is strange that they should speak Catalan.

Experience with terminology
It is also important to consider the experience of the typical user who will be using the software. For example children in Britain and America have typically had far more contact with computers, even if they are games machines, than children in Bulgaria and this may affect adaptation. People unused to computers can find it difficult to cope even with translated text because the translated terms refer to concepts which are unknown to them, and in this case translating terms does not get you very far. If the mass of users are children it is wise to use the simplest possible translated terms, and explain them clearly to make sure they are understood, even at the cost of using more words.

Of course those people who are at ease with English can continue working with the original versions, but it is important to bring professional computer users into the localisation process, as it is they who have the most influence on the development of new terminology. If they are not satisfied with the translated terms they should be challenged to produce better equivalents.

Getting tone and register right.
It can also be tricky getting the tone and register right for other languages. English computing terminology is full of jokes which disappear for speakers of other languages. Words and acronyms like bug, floppy and WYSIWYG, all have a light hearted tone for English speakers, but for speakers of other languages they can have the same solemn formality that medical Latin has for English speakers. It is necessary to make a conscious effort if a lightness of touch in the translation is to be maintained. The authors of this paper like software to be entertaining so they have included a number of visual, textual and aural jokes. English humour is often considered dry and ironic, and as a lot of humour is dependent on puns, translation is difficult. Often the best tactic is to use a phrase with a completely different meaning but an equivalent humorous value. When adapting humorous elements in a program it is important remember that different cultures expect humour in different situations, and that what is appropriate in English may not be appropriate for other cultures. As a result it is vital not simply to work with a good linguist, but also with someone who works with the learners who will use the software.

It is also worth considering the level of sophistication of the users, since if they are unused to the interface, humour may confuse them instead of making them feel comfortable. For example, at one point in the Work Rooms a sound is played to the learner saying "Warning, this is a talking computer." Adapted into another language this could be quite misleading for a user who has had no previous experience of computers. This might well be the case in Bulgarian schools and homes, where access to computers of any kind is limited and ideas of what computers can achieve may be formed from science fiction as much as direct experience.

Visual issues

Icons and metaphors
There is no doubt that visual symbols are more widely understood than words, and so it might be thought that working with a graphical user interface, as in the Work Rooms, would help the translator. Unfortunately using visual symbols frequently only serves to lull one into a false sense of security. The interpretation of a symbol depends on the context, and when the context changes the meaning is liable to change or disappear too. British residents might well imagine that an image of the sun shining away in the corner of a screen would conjure up feelings of pleasure and satisfaction throughout the world, but the inhabitants of hot arid countries would be more likely to interpret the same image as a threat of pain or death. A picture of a person tapping her or his head indicates intelligence or stupidity depending on the cultural context, and when many people count on their fingers they inadvertently produce the two fingered British insult, while the American hand signal for OK means "three" to many Africans.

Symbols
In testing the Catalan and Bulgarian versions of the Work Rooms we realised that the tick symbol was completely unknown to the children, and we had to change it. Ironically "OK" proved to be a good international substitute. Differing conventions for numerical symbols also caused confusion, as in Spain 1.000 represents one thousand, while the comma in 1,593 represents a decimal point - the reverse of the English situation, and this meant that the code which performs calculations had to be altered.

Reading the screen
The layout of the screen is not interpreted in the same way by all cultures, as the conventions for reading Latin based languages encourage Europeans to read graphic images from left to right, with a secondary movement from top to bottom. If we depict an activity to be followed on the screen, this is how we will assume we should interpret its development in time Those who have been brought up with Arabic or Chinese scripts, for example, have a different set of assumptions, which may need to be taken into account.

The Work Rooms Experience

The issues which we have dealt with so far apply to anyone who sets out to adapt software for another culture. Now are considered the characteristics of the technical environment which the Work Rooms were developed in, and its implications for localisation. This technical environment was the Apple Macintosh System 7 using HyperCard as the main development system.

Resources and the human computer interface

One of the strongest features of programming for the Macintosh is the division of files into a resource fork and a data fork. Elements such as dialogue boxes, menus, graphic images, fonts and sounds etc. can be stored in the resource fork of a HyperCard file, and can be edited using a Macintosh utility called ResEdit without interfering with the functioning of the software or the system as a whole.

Because the resource management system is hierarchical in nature, overriding standard system resources is possible, delivering great flexibility and local control of the user interface.

While ResEdit is a powerful program which has to be used with care, using it is a great deal easier than having to edit and recompile source code. Its strength is in the separation of code, data and user-interface which allows non-programmers to alter the user-interface in a clean and controlled manner - ideal for low-cost adaptation to another language and culture.

The design and development environment

The Work Rooms are written in HyperCard, a Macintosh specific high level environment. The ease of use of HyperCard's programming language, HyperTalk, facilitates the adaptation process, but the translator can still encounter difficulties.

There are three main problem areas : screen design, software structure and font technology. Our main recommendations are grouped under these headings, and in each case it is clear that planning for translation can make adaptation easier and cheaper.

Screen Design
- Interface elements often use a mixture of graphic images and words. This can present problems if a word or two in the original has to be translated by a number of words or even a sentence in the target language, so it is important to leave plenty of room around the screen elements.

Software structure

- In HyperCard you can give names to buttons, automatically show the names on screen, and refer to the names in the program code. This combination is very useful, but when the name of a button is changed in translation, scripts which refer to it cease to function. One solution is to switch off the "show name" option and replace the text with a separate HyperCard object - a text field. This clearly complicates the task when implemented as an afterthought. Consequently we recommend that text associated with a button is stored in a field, which can be edited without interfering with the button's function, and which can display non-Latin fonts. This creates a little extra work for the original author, but saves a great deal of trouble for the translator.

- HyperTalk is object oriented, and it is not always obvious to the translator where in the fragmented program code a particular piece of text or dialogue box is being generated. Solutions to this problem include storing text messages in hidden fields so that the translator has one place to edit. This cannot help in every situation because on occasion the message is constructed in a context sensitive fashion by the program code. Nevertheless, static text messages can be dealt with successfully in this way.

- Care has to be taken when designing purpose-built fonts. One of the Work Rooms, Broken Calculator, uses a custom designed font to show numerical digits in liquid crystal display style (see Fig. 4) and the message TOO BIG on the calculator display when a number is too large to be shown. As versioning the program for other languages was not envisaged when the program was written only the necessary letters were included in the custom font, and the translators had to create more such letters to display a translated message.

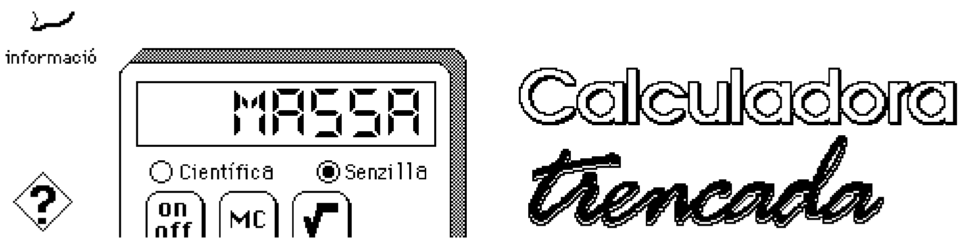


Figure 4 : The custom font to display LCD numerals and messages in the Broken Calculator

Font technology

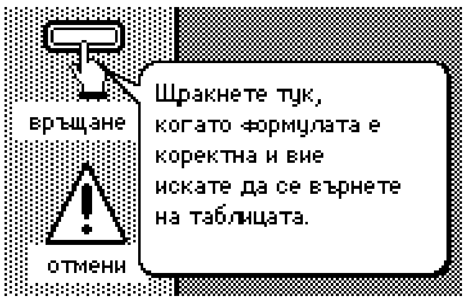
Apple Computer is making great efforts to cater for applications which support multiple languages and alphabets. In particular it is possible for the user to switch between different language systems within the same document for both display and text entry.

To a large degree this capability can be automated. For example the latest version of QuickTime, system software for handling dynamic media such as video, is able to recognise the language which the computer is using, and if it supports that language it automatically switches to it.

This development is particularly significant because it enables the Macintosh to manipulate other than Latin scripts. This is particularly important for the authors as we believe that it is important that learners can represent their knowledge and ideas on the screen, and the new "World Script" technology permits Bulgarian users to enter their work in text fields in the Work Rooms in a Cyrillic font or, critically, in a mixture of fonts.

Unfortunately, applications such as HyperCard often lag the technology innovations which Apple deliver and in this case, HyperCard was not able to accommodate Cyrillic fonts completely. We had to write a new extension command (XCMD) to enable dialogue boxes created in HyperCard to use other than a standard font or style. Another solution tested was to renumber a Cyrillic font resource and add it to stacks to override the system font resource.

We have also written an extension command to display balloon help within our HyperCard programs. The original version was written to function with only one font, Geneva 9, and for the Bulgarian version of the Work Rooms it was necessary to create a new version to deal with a Cyrillic font (see Fig. 5).



Щракнете тук, когато формулата е коректна и вие искате да се върнете на таблицата.

Figure 5: A balloon help message using Cyrillic.

Checking

When the adaptation is finished it is essential to exhaustively check the software's program code. There is, however, no substitute for lengthy testing in the country where the adaptations are to be used with actual learners in the contexts in which it is anticipated to be used.

Taking our own medicine

In developing the Work Rooms software we have learnt many lessons. A new development, Carnaval des Animaux which is a participative environment for capturing and illustrating musical "snippets", has been designed to support a range of languages and cultures and will offer multilingual support in the initial version. The software will both adapt to the system environment it finds itself in but also allow the learner the choice of which language messages will be displayed in (see Fig. 6).

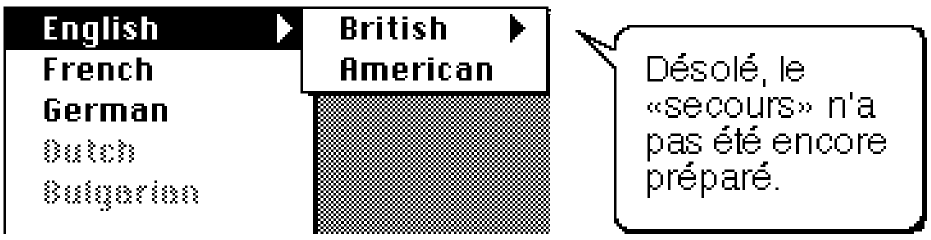


Figure 6 : A choice of languages for messages available in Carnaval des Animaux

In addition, text in general has been reduced to a minimum and graphic and iconic elements, explained with context-sensitive balloon help make up the majority of user interface elements.

Conclusion

There are major benefits, socially, educationally and financially for all concerned with educational software from the adaptation of educational software.

With careful planning the process of adapting an application in HyperCard can be made simple enough for a translator with only a superficial knowledge of software development. For this to go smoothly it is vital to:

- plan for adaptation when the program is designed;
- adapt in an international team so that :
    - the objectives of the authors can be understood by the translators;
    - the curricular and cultural needs of the target community can be understood by the authors, who may then be willing to make changes to their program;
- develop contacts at a personal and institutional level which will permit thorough testing with users who speak the target language.

We hope that English speakers will get used to the idea that software can be translated, and that the computer can work with different alphabets, so that information technology does not unify all cultures but acts as a bridge between them. We can testify that the international contacts which translation work generates are enriching for the translation teams, and we hope that projects such as ours will help others to gain new perspectives on the world we live in.

References:

[1] Van den Brande L., R&D on learning telematics in the European Community, *Journal of Computer Assisted Learning* **9**, (1992)

[2] Cox M. and Bosler U., Training Authors to Develop Educational Software through the Adaptation of Foreign Software. *Computers Educ.* **16**, 71-76, (1991).

[3] DeVillar R.A. and Faltis C.J., *Computers and Cultural Diversity: Restructuring for School Success*. University of New York Press, (1991).