

A Reference Model for Software Localisation

S. P. Mudur
Concordia University,
1455 de Maisonneuve Blvd. West.,
LB 903 - 3, Montreal, Quebec,
Canada H3G 1M8
Email : mudur@cs.concordia.ca
Rekha Sharma
Infosys fellow,
CDAC, Juhu, Mumbai 400 049
India
Email : rekhag@ncst.ernet.in

Abstract

Without urgent and satisfactory localisation of digital systems, it is difficult to envisage any significant impact of Information and Communication Technologies (ICT) in less literate nations such as India, where over 92% of the population cannot communicate in English or in any of the other human languages supported today in these new innovation products. Recognizing this need, most of the major software manufacturers are putting in considerable efforts in creating and releasing internationalized versions of their popular product offerings, with the underlying assumption that internationalized software is easily localized. Localisation is itself an evolving activity in ICT. There are various degrees of localisation, ranging from simple translation of hardcopy documentation to the accommodation of culture specific aspects in product usage. Current internationalisation efforts are at best limited to enabling local language input and display. A few also support local language versions of menu items, dialogue box texts etc, largely through a table driven translated substitution of textual elements. Higher degrees of localisation would however require more sophisticated handles within the software product. Much like software development, localisation is a very human intensive task covering the entire life-cycle of a digital system – from conception to retirement. A reference model for software localisation is needed showing the different phases, activities and tools required to carry out localisation to a desired degree. In this paper we propose one such reference model, that could potentially be used during internationalisation by software manufacturers to decide on the level of localisation their product would support and also by localisation engineers during the actual localisation process.

Keywords

Internationalisation, Localisation process, Localisation engineering, Localisation reference model.

1.0 Introduction

Like electricity and post industrial revolution advances in manufacturing, new innovations in information and communication technologies (ICT) have the potential of transforming societies and thus opening up enormous possibilities for development. The information revolution is surely and certainly proceeding, even if at different paces, all over the world. Its effects are deeply felt in all sectors of economic and social life. However, effective participation of developing countries in the emerging global networked economy remains a challenge. Free and abundant flow of information is an important contributor to liberal markets, better human resources, and an open society. Developing nations are yet to explore and exploit these new digital opportunities presented by Information and Communication Technologies (ICT) to foster rapid growth and benefit the poor. There are many factors contributing to this –national policies, infrastructure, costs, and technology diffusion problems, just to name a few.

Digital literacy and not just literacy forms the basis, if this technology has to have an impact on any society. In a less literate country like India, where over 92 percent of the population does not know the English language nor any of the languages supported by digital systems, substantially increasing the number of digitally literate is a virtually impossible target with the type of digital systems that are available from around the world. Clearly localisation of these digital systems is essential.

Software localisation can be defined as the process of creating or modifying a software product so that it can be easily used in the local language, culture and environment of a particular region. For example, at the very basic level, the localized system must support the character set of the local language and must be configured to present numbers and other values in a format common to that region. The computer dictionary [1], which defines localisation as “*the process of altering a program so that it is appropriate for the area in which it is used*” lists the following requirements:

- A computer must be capable of displaying the user’s native language, character set, and notations (such as currency symbols).
- The user interface and documentation must be in the user’s native language in a way that is understandable and usable.
- A system must match the user’s cultural characteristics. It must accommodate the way business is conducted and the way people communicate in those regions.

Internationalisation (i18n) is the term used today for getting a software product ready for localisation (l10n) to a specific user community[2,3,4,5,6]. A data structure known as the “locale” is used by many internationalized software platforms to support the localisation process. The localisation process itself [7,8,9] varies from one project to another depending on the locale, the platforms on which the software application will run on, resources used, size of the project and timeframe requirements. The process also differs by localisation strategy and the i18n support already provided. Although, the need has been felt all over the world, localisation is still in its nascent stages. The concept itself is not well known among various software developers and users.

All the current localisation approaches are very difficult and effort intensive. The reasons for this are listed below [10].

- Language and cultural conventions are difficult to translate.
- Considerable variation in localisation requirements
- Lack of suitable automation support
- Lack of standards
- People with multidisciplinary skills are required

All the above have contributed to making the localisation task very laborious, cumbersome, tiring and time consuming. Coupled with the fact that payoffs in the developing countries could have more latency, software manufacturers, who are dominantly from the western world, do not find it attractive to invest vigorously in this field.

The result is that until now most localisation attempts have been restricted to text level localisation. Usually cultural aspects are ignored, because of the extensive effort that may be involved in representation and proper translation of various cultural conventions, or due to technical difficulties in product modification [11].

2.0 The Proposed Reference Model

The localisation process has been identified based on the following degrees of localisation in order of increasing complexity and effort:

L0: Translate nothing

L1: Translate hardcopy documentation and packaging only

L2: Enable the code

L3: Translate software menus and dialogs

L4: Translate online help, tutorials and README files

L5: Translate culture specific features

The degree of localisation determines the amount of translation and customisation necessary to create localised editions. Full localisation ranges from translating nothing to shipping a completely translated product with customized locale specific features. The final decision for partial or full localisation is an economic one and depends upon the level of investment one is willing to make.

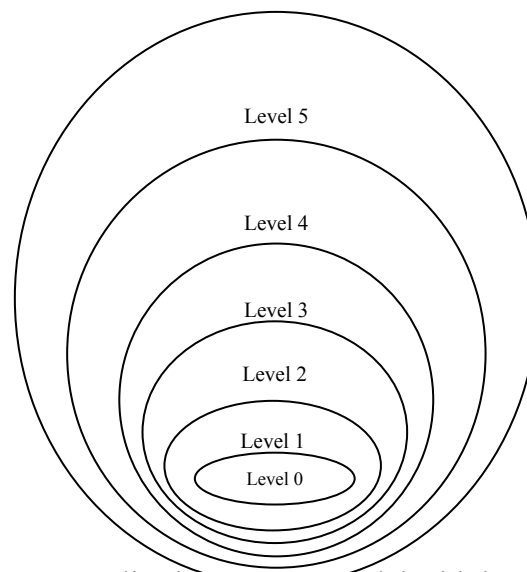


Figure 1: Localisation process model with levels

Depending on the degree of localisation desired, we can define different processes (See Fig. 1). Moving down from level L0 to L5 increases the cost and risk involved. L0 is the trivial case. At the lowest level (L1) we can localise only hardcopy material of the software. For example the electronic instruments like television, phone come up with localised manuals. All the screen-shots etc may be in English but the text is localised in many languages. Similarly software's hardcopy manuals can also be made available in local languages for increasing the use of the product for a larger community.

The next level (L2) requires the software to have capabilities that enable a user to read, input or edit in local language. This is a very critical activity. Actual code has to be developed here. Unless the platform has a localized version with support for these language level tasks [12,13], source code level changes may be needed in order to support this activity. Once level L2 is achieved, L3 involves translation of text into local language. In addition, it is important that the software internationalisation did support the use of different resources in the form of tables of text items [8], so that the software interface and interaction with the software does become possible in local language. The second last level involves translation of all the software related documentation, both hardcopy and online versions. Depending on the volume of documentation, this can be one of the

most effort intensive activities. And at the last level (full localisation), culture specific features are also embedded. This will involve customisation of the application according to locale specific preferences about colors, icons, style and placement of windows on the desktop, etc.

3.0 The Localisation Process Model

Localisation of the product to any desired level can be performed using waterfall, incremental, spiral or combinations of traditional software process models [14,15]. We can define the tasks, activities and tools involved in the localisation process according to levels, along with the input and output of each activity. Below we give illustrate this by discussing the detailed activities etc. at some of the levels.

L0: Translate nothing: First option is not to enable or localise the software product at all.

L1: Translate hardcopy documentation and packaging only: At this level localisation includes translation of printed manuals, installation guidelines, collateral material, quick reference cards, and marketing pieces. Since the software has not been localized, screen-shots and other images will remain in the original language; only the instructions given to the user will be translated into local language.

This level involves little development cost - at most, developers will be called upon to explain technical issues to the translators.

The abbreviations for different persons involved are given in the following Table 1, and the process is shown in Table 2.

<u>Persons Involved</u>	<u>Abbreviation</u>
Localisation Engineer	LE
Localisation Manager	LM
Translator	TR
Testing Engineer	TE
QA Specialist	QS
Proof-reader	PR
Linguist	LG
Web Designer	WD
Web Manager	WM
Web-site Builder	WB
Client	CL
Font Designer	FD
Desktop Publisher	DP
Localisation Vender	LV

Table 1: Localisation team

<u>Tasks</u>	<u>Activities</u>	<u>Tools</u>	<u>Persons involved</u>
1. Plan for the localisation project	1.1 Collect the material (obtain the hard copy) to be translated with specifications (deliverable, platform, version, style sheets, font, etc)		TR

	1.2 Analyse the content needs to be translated.		TR, TM
	1.3 Estimate the cost based on the amount of translation (word count), no of translators, cost of printing material	#	TM
	1.4 Assign the translators and explain them the technical issues involved in translation and provide guidelines.		
2. Translate documentation and packaging	2.1. Open a file to be translated in the editing system. (If the soft copy is not available then the material should be typed in first.)	Alchemy CATALYST,	TR
	2.2. Collect reference material like dictionary etc.		TR
	2.3. Create a glossary of commonly used technical words in the documentation.		TR
	2.4. Check the spellings, leverage translations, create translation memories.		TR
	2.5. Perform on-line (automatic) or offline (manual) translation of material.		TR
	2.6. Recollecting all the translation done by different translators		TR
	2.7. Printing translated material.		DP
	2.8. Proofread the translation. Translator as well as experts must do proofreading.		TR
	2.9. Perform T/E/P cycles, till one can get a freezed version.		TR
3. Test the translations	3.1. Translations should be correct context-wise.		TR, CL
	3.2. Translation should be according to target audience.		
	3.3. Testing complete translations for correctness		
4. Hand-over the translated material (printed or softcopy) to the client.			

Table 2: Level 1 Localisation process

L2: Enable local language code

Enabling the local language code does not involve any translation cost, but involves development cost. The source must be altered to handle input, layout, display, editing, and printing of local language text. Screen savers multimedia titles and other programs that do not allow the user to enter or edit text do not require enabling. The activities at this level are crucial and also difficult.

<u>Tasks</u>	<u>Activities</u>	<u>Persons involved</u>	<u>Tools</u>
1. Provide additional services to support commonly used standard (National /International) for character encoding.		LE	
2. Develop local language fonts and	5.1. Study BDF, true type,	FD	Fontographer,

display layouts.	<p>open type font technologies</p> <p>5.2. Decide or find out the purpose of designing the fonts. Objective may be on-screen display, Printing or projection.</p> <p>5.3. Decide the sign list (no of glyphs) in the glyph chart.</p> <p>5.4. Decide the design primitives, i.e. the common properties of all the glyphs and shapes in the font. These design primitives takes care of look, mechanical shapes, calligraphy, language characteristics, clarity, beauty and purpose.</p> <p>5.5. Select appropriate tool, which gives sufficient freedom to edit the letters.</p> <p>5.6. Testing the fonts. This includes the testing of conjuncts, Matras, distance between letters and Matras and between two different types of letters.</p>		Fonty-dev Graphite, Uniscribe, Addtable Ttoasm Vtt Otlsign
3. Design the shaping engine for local language text display.	<p>6.1 Syllable breaking and repositioning</p> <p>6.2 Character / glyph re-ordering</p> <p>6.3 Shaping context extraction</p> <p>6.4 Glyph composing and positioning).</p> <p>6.5 Test the shaping engine using font and language rules (Some testing tool can be developed)</p>	LE	Volt
4. Create and manage Locale definition file. (An interactive interface can be made to manage the locale definition file)		LE	
5. From the source code, find out the functions, methods, libraries responsible for input, display printing, etc. Refine the methods to enable local language.	<p>8.1. Input methods (Design new keyboard layouts schemes)</p> <p>8.2. Rendering methods</p> <p>8.3. Editing methods (Curser movement and hit testing, insertion and deletion of syllables)</p> <p>8.4. Printing methods</p>	LE	
6. Integrate shaping engine with the software and test (with developed fonts).			FreeType library

Table 3: Localisation process for language enabling

Very few tools are available for the above activities. Experience in localisation projects matters a lot at this level, since highly technical and strategic decisions have to be made by a localisation engineer.

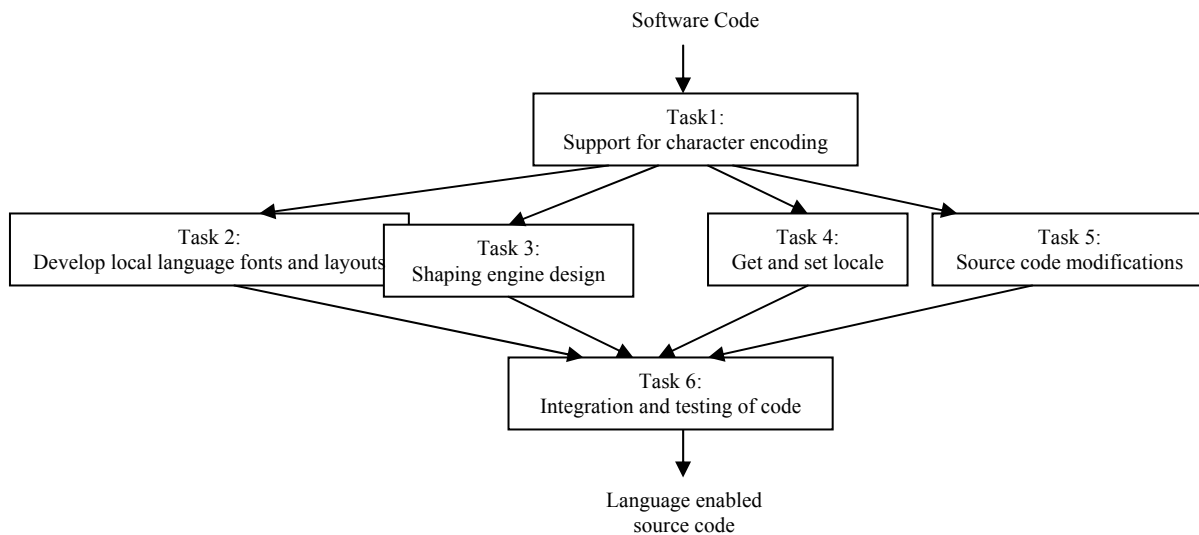


Figure 2: Sequence diagram for L2

L3: Translate software menus and dialogs

If an English-language edition of the product exists, people sometimes only need the user interface to be localised even if they do not opt for full localisation. It includes translation cost. As a general rule, the more involved the translation, the more questions developers must answer and the more bugs developers must fix.

Major task at this level is the translation of a huge amount of message strings, which are part of GUI. The process model is an extended version of L2. Table 4 shows the input/output for each of the tasks.

L4: Translate online help, tutorials and README files

More cost is involved at this level because of the amount of the translation, and possibly development of special pieces of code. For most localisation projects, online help is the largest component. Help files and online manual have taken the role of printed manual. Once an acceptable version of the software to be localized is available, translation of on-line help and documentation can start. Automation support is urgently required [16,17].

L5: Translate culture specific features

This level require one to analyse the cultural conventions of the target user community. In more competitive markets, customized features are needed. For example drop down menus may be unnatural for a specific culture and they may want to retain windows at one corner of the screen. So it has to be taken care of while designing the GUI for the product.

Tasks	Input	Output
1 Analysis of Received Material	<ul style="list-style-type: none"> • Software • Compiler 	<ul style="list-style-type: none"> • Bug report • Technical information about the project
2 Budgeting and Planning	<ul style="list-style-type: none"> • 	<ul style="list-style-type: none"> •
3 Glossary Translation or terminology setup (Design of product specific glossary)	<ul style="list-style-type: none"> • Product glossary • Source to target language dictionary 	<ul style="list-style-type: none"> • Translated product glossary
4 Preparation of localisation kit.	<ul style="list-style-type: none"> • 	<ul style="list-style-type: none"> • L10n kit
5 Translation of software	<ul style="list-style-type: none"> • Product glossary (Translated) • Dictionary and other reference material • Resource files (Source Files) • Resource compiler 	<ul style="list-style-type: none"> • Translated resource files (Target language) • Glossary (translated) of technical terms for future use • Translation memory
6 Processing Updates	<ul style="list-style-type: none"> • Translated message files (Resource files) 	<ul style="list-style-type: none"> • Updated source files (software)
7 Testing of Software	<ul style="list-style-type: none"> • Translated message files (Resource files) • Updated source files (software) 	<ul style="list-style-type: none"> • Testing updates • Testing updates
8 Product Quality Assurance and Delivery	<ul style="list-style-type: none"> • Translated message files (Resource files) • Updated source files (software) • Client requirements 	<ul style="list-style-type: none"> • Bug reports • Localised product

Table 4: I/O for activities for L3 Level Localisation (**Input:** Unlocalised Resources **Output:** Localised Resources)

Umbrella Activities

In addition to the above activities, a set of *umbrella activities* are also defined as part of the reference model. These umbrella activities persist across the entire software process. These umbrella activities include:

- *Software project management*
- *Content Management*
- *Documentation*
- *Software quality assurance*
- *Reusability management*
- *Risk management*

4.0 Conclusion

Localisation is a very effort intensive activity and requires a systematic approach covering the entire life cycle of the localized product. As a result, most technology providers do not find it worthwhile investing in the creation of truly well localized digital software and systems. In this paper we have proposed a reference model for localisation based on degree of localisation, identified the major tasks and activities for the localisation process at different levels. It is clear that automated support is essential if localized versions of new innovative products have to hit the developing country markets with minimum latency. It is also important that market research should be conducted to determine the extent of localisation which makes sense, taking into account the reality that different levels of

localisation can be considered, each with different costs and different effects on attainable market penetration.

Acknowledgements

Much of the above has been gathered from major localisation projects that the author(s) were involved in. These include the localisation of Microsoft Windows™ environment and also the Linux environment. Both projects were acrid out when the first author was at the National Centre for Software Technology (NCST) in Mumbai, India. The authors are grateful to their respective employers for their support in the writing of this paper. The second author is registered for her Ph.D. at ITW (Institute of Technology for Women), and wishes to acknowledge support from Infosys Ltd. in the form of a doctoral fellowship. We also thank colleagues at NCST who were involved in these projects and contributed significantly to their success.

References

- [1] Microsoft, *The Computer Dictionary, Fifth Edition*, Microsoft Press, 2002
- [2] Kano Nadine, *Developing International software for windows and Windows NT*, Microsoft Press, 1995.
- [3] Turnbull Stephan, *Alphabet Soup: The Internationalisation of Linux*, Linux Journal, 1999.
- [4] Sun Developer, *Internationalisation*, <http://soldc.sun.com/articles/i18n/>
- [5] Karat J., Karat C. M., *World-Wide CHI: Perspectives on Design and Internationalisation*, SIGCHI, 28(1), 1996.
- [6] Esselink B., *A Practical Guide to Software Localisation*, John Benjamins Pub., 1998.
- [7] Zinc White Paper: *Developing Internationalised Applications with Zinc*, http://www.windriver.com/zinc/html/zinc_int_white.html
- [8] http://www.conveysoftware.com/en/serv_localiz.html
- [9] Microsoft: Global Software Development, <http://www.microsoft.com/globaldev>
- [10] S. P. Mudur, R. Sharma, Shroff K., Rajan P., *An Integrated Environment for Software Localisation*, Proc. ICCC 2002, ICCC Press, Washington, 2002.
- [11] S. P. Mudur “*On the need for Cultural Representation in Interactive Systems*” Frontiers of Human-Centred Computing, Online Communities and Virtual Environments, Ed Rae Earnshaw, Richard Guedj, Andries van Dam and John Vince, Springer-Verlag London, pp299-310, 2001.
- [12] SP Mudur, N. Nayak, S. Shanbhag and R_K_Joshi, 1999, *An Architecture for Shaping of Indic Scripts* Computers & Graphics, 23(1), p7-24.
- [13] S. P. Mudur, K., Gaur R., Kumar V., Y. Ketkar Y., *Indic Script processing under X11*, Proc. of the Symposium on Translation Support Systems 2001, IIT-Kanpur India, 2001.
- [14] Pressman R. S, *Software Engineering*, McGraw-Hill, 1997.
- [15] Erica L. Young, *A Framework for the Integration of Internationalisation in the Software Development Process*, MA Thesis, South Dakota Univ, 2002.
- [16] <http://www.simultrans.com/seminars/seminar199907.htm>
- [17] R. Sharma, Shroff K., Mudur S.P., Rajan P., *Development of A Bilingual Electronic Glossary For Automated Assistance In User Interface Localisation*, Proc. of the Symposium on Translation Support Systems 2002, IIT-Kanpur India, 2002.