Towards a Model of Integration of Underserved Cultural Factors in Software by Reverse Localisation: Case Study in Yemba Culture

Mathurin Soh(✉)

Department of Mathematics and Computer Science,
University of Dschang, P.O. Box 67, Dschang, Cameroon
mathurin.soh@univ-dschang.org

Abstract. This paper aims at contributing towards an empowerment of underserved culture through computer applications. It is inspired by the conviction that successful strategies can lead to balanced chances for all cultures in the context of computer applications. It raises and investigates the problem of cultural divide observable in computer applications and pave the way to the preservation of underserved cultures with such tools. We find that reverse software localisation approach as defined by [1] can help under-resourced cultures not to be engulfed by the dominating Western way of building computer applications. An approach to integrate underserved cultural factors in computer applications is provided and consist of investigation of cultural markers, followed by translation or adaptation, implementation and evaluation by end-users. As a case study, we investigate the cultural localisation of editor for the Yemba culture in Cameroon. It gives access to the use of a text editor to users fluent in Yemba culture. The adaptation of most computer software to non-native environments does not always capture the features of the target culture. Despite some trade-offs, this work has a strong symbolic value in the empowerment and preservation of the Yemba culture, and for underserved cultures.

Keywords: Cultural divide · Cultural factors · Localisation · Editor
Underserved culture · Yemba

1 Introduction

Computer applications and Information and Communication Technologies (ICTs) tools have become daily useful and are the centerpiece of the global information society. These software are viewed as artifacts which interact with cultures of societies in which they function [2]. Kersten et al. [2] prove that like any other product, these applications contains embedded designer's cultural values and objectives. The same authors specifies that some of the embedding occurs unconsciously, inherited via the cultural programming of its human creators; other parts of it are intentional via design requirements explicitly obtained by researching its target markets [2]. End-user's who understand the language and culture of computer applications are able to use them, while other people might be entirely locked out or unable to fully enjoy the benefits these applications brings [3]. This way of designing and programming computer applications result in a cultural digital divide, observed in ICT tools and computer applications between underserved cultures and western ones. The greatest challenge remains on how computer applications globally built, but locally used [4], could be adapted to fully support end-users' cultural features and contents, to help underserved cultures not to be engulfed by the dominating western way of building interfaces. Globally, this can be solved by means of software localisation, which is the adaptation of software to linguistic, cultural and technical requirements of a target market. Localising user interfaces has been proven beneficial for both user satisfaction and work efficiency [5]. Besides, the local population has the right to practice and refresh its own cultural traditions, customs and knowledge while using technologies and computer applications in its own terms and ways. Yacob [6] offers a broad interpretation that defines the localisation as: *the transfer of cultural consciousness into a computer system, making the computer a natural extension of the society it serves.* But the adaptation of computer software to non-native environments does not always capture the features of the target culture. It is emphasized in [7], that successful software systems must be written so that adapting them to a particular culture can be done easily.

This paper aims at raising and investigating the problem of preservation of underserved cultures through computer applications and ICT tools which have become daily useful tools. The matter of this work is to find a model to integrate underserved cultural factors in computer applications, information systems, in order to preserve underserved cultures through such tools.

In the remainder of this paper, the Sect. 2 presents the cultural digital divide problem. Because the digital divide is as a complex and dynamic phenomenon, this section specifies the software engineering based counterpart. The Sect. 3 discusses related works on capturing and integrating cultural features of underserved cultures in computer applications for the purposes of alleviating cultural digital divide. The Sect. 4 is about the reverse software localisation approach. The Sect. 5 provides the methodology. It specifies the paradigms, the processes and the technologies. In the Sect. 6, we apply it to the case of preservation of Yemba Culture. Its first part specifies major features of Yemba culture. In its second part, the reverse approach combined to our methodology is applied to the reverse localisation of a text editor in Yemba. The third part presents the results and the related open issues. We end with a conclusion in the Sect. 7.

2 The Cultural Digital Divide in Software Development

The term *culture* has many definitions and does not have a unified definition, but has been modified depending on different research disciplines [8]. Hofstede [9] defines it as "*the collective programming of the mind which distinguishes the members of one group or category of people from one another*". Considered as the software of the mind [9], culture is thus the integration of human behaviour that includes attitudes, norms, values, ideas, beliefs, rules, material dimensions, actions, communications and groups (ethnic, religious, social, etc.). Composed of overt and covert factors, culture is like an iceberg with invisible and visible parts.

Computer applications' views are designed as a matter of taste as preferences vary from person to person [8]. These preferences can be found deeply-rooted in culture. For example, software applications like text editors, spreadsheets have managed to become an every-day tool for the vast majority of the population. Existing ones have, if any, a Western culturally biased interface [10]. The use of such tools is more difficult for users who are from other languages and cultures, resulting in a cultural digital divide. In order to compute a cultural product, it is recommended in [11] to pay attention to, amongst others, the following characteristics: text direction (left-to-right, right-to-left), date and time formats, numeric and currency formats, icons, buttons and colours, etiquette, policy, tone, formality and metaphors.

The currently ongoing cultural divide in computer applications may accelerate the inequalities among cultures in the digital world [11,12]. The idea of the cultural digital divide refers to the growing gap between the underserved cultures, which are not used in computers or the internet; and the privileged cultures, especially western ones, whose features are used in most computer applications, tools and Internet. The preservation of cultural factors has become at present a subject of fundamental importance for most communities. To tackle such a complex subject, the use of informatics is required, since the traditional methods, based especially on paper documentation, are absolutely inadequate [13].

These issues can have causes in the interplay of software development and integration of underserved cultures's factors or aspects. In fact, during development, designers and developers inconsciously integrate their own cultural values, norms, practices in the process and/or the final product. And the end-users who understand the designers' culture are able to use the product while others might entirely be locked out; some unable to fully enjoy its benefits, some are quite excluded, or for others their culture is progressively absorbed. This result in a more and more cultural digital divide. For example, considering an overt cultural factor like language, statistics shows that about 77.9% Internet users of world total population speaks ten languages [14]. These languages are respectively English, Chinese, Spanish, Arabic, Portuguese, Japanese, Malay, Russian, French and German. Knowing that language is the vector of culture, we can conclude that cultures not used in ICTs and Computer applications are likely to be underserved. Software development could help these cultures in integrating their own overt or covert cultural factors in computer applications in order to preserve them in such tools. Thus, solving the problem of cultural digital divide turns into finding an approach to build computer applications that integrate end-users' cultural factors.

3 Related Works

The cultural digital divide has received too little attention in scientific community. Research increasingly shows that one of the essential ways to attack digital inequalities is by addressing the fact that technologies are always created with cultural biases built-in that limit their use [13]. This means that the divide will be lessened only when, in addition to providing basic access, we address seriously cultural differences and the differences in power that come with them [15]. In correcting this bias, the focal point of research should be shifted towards the people. We should make computing technology available, understandable, and participable for everyone regardless of culture, gender, age, income, language, degree of disability, or ethnicity [4].

The preservation of cultural heritage objects through the use of computer modeling techniques has attracted considerable attention in computer graphics, geometric modeling, and virtual reality communities [16]. This problem has less attention in software development area. Despite the cultural influence of the designers own cultural preferences on the design process, there have been limited studies examining this.

From personal computing till cultural computing, there exist many attempts to go closer end-user's culture in computer applications. Cultural computing is more than integrating cultural aspects into the interaction. It is about allowing the end-user to experience an interaction that is closely related to the core aspects of his/her culture. In a way that let him engage with an augmented reality using the values and attributes of his own culture. As such, it is important to understand one's cultural determinants and how to render them during the interaction [17]. Research in this domain of cultural computing is concentrated between three critical domains: numerical arts, virtual worlds and technology adaptation. The latter make use of localisation, necessary to the adoption of technologies by others cultures.

In [13], Bartolotta et al. deal with the cultural heritage inventory selected according to historical urban centres; archaeological sites; extra-urban architectonic assets (castles, towers etc.). To handle the great deal of information related to this heritage, a suitable Geographic Information System (GIS) is designed, aimed at having different information (texts, photos, drawings, talking or music sounds, videotapes) and easy to be integrated and updated in future.

According to [18], ICTs has been applied successfully to numerous remote Indigenous communities around the world. The greatest gains have been made when requirements have been first defined by Indigenous members of the community, then pattern matched to an ICT solution. The same work shows the potential uses of ICT in key fundamental areas to the continuing presence of a culture. It uses GIS, multimedia clips, digital document archives for the preservation of culture. But, because culture is something alive and ever-changing, its preservation cannot be achieved by ICT alone; it requires the spiritual element behind the history to be actively reinvigorated into a community to make its presence felt in a long-lasting manner.

Other attempts to solve the problem of the digital divide have focused on a shallow interpretation of technical literacy as simply learning computer programs, unaware that technological forms are culturally shaped and need to be reshaped to fit a wider variety of cultural styles and forms [15]. One of the best ways to do this, is encourage and provide technical training for people who have lived experience as members of underserved groups. These culturally competent, technically savvy individuals can then work as facilitators for marginalized communities to empower them to represent themselves in digital media on their own cultural ground via their own cultural forms.

The realization of a GIS and ICT tools means a great step forward in the field of cultural heritage inventory and handling [13]. But preserving culture takes more than ICT [18]. The advantages of this ICT tool is clear enough also from this work, however limited it is. The inherent problem with ICT is that while it is good at preserving tangible knowledge it has difficulty with how to treat tacit knowledge. ICT output, no matter how well represented, is usually one-dimensional [18]. Modelling cultural factors and its handling within computer applications console based or Internet based need the development of techniques of visualization, like the VRML (Virtual Reality Modelling Language) graphic language. In fact, they are absolutely more fascinating than traditional two-dimensional representations, and the possibility to query and explore the system, through Internet too, make imaginable a marked increase of the GIS applications in the field of cultural heritage [13].

One drawback of such approaches is that they cannot effectively capture and tackle the complex nature of users' cultural requirements in the target culture. But, according to Kersten et al. [2], there is no solid theory that links software and culture, or the way ideas and values are implemented in software. Such a theory is required and needs to go beyond the consideration of the surface manifestations of culture that have been widely accepted in software internationalization methodologies and address the core components of software that, we believe, influences our ideas and values [2]. In [19], Anacleto models cultural differences by comparing knowledge bases of common sense statements. This work points out that this kind of knowledge can help computer systems to consider cultural differences. [8] presents an approach to factoring culture into user models, by introducing the cultural user model ontology (CUMO), which describe how and to which extend it can accurately represent the users cultural background. Reinecke et al. [5] propose a new approach to localisation by modeling the users culture according to its understanding in cultural anthropology. Contrasting this view with cultural influences on user interface perception and preferences, they obtain an intersection of aspects that need to be included in a cultural user model, and deduce which user interface aspects have to be adaptable.

4 Reverse Localisation Approach for the Integration of Cultural Factors

With the cultural imperialism of western computer applications, preserving cultural features of underserved communities turns to modelling indigenous cultural factors and integrating them into nowadays computer tools and applications [20]. The suited approach is named reverse localisation defined by Schaler [1]. Despite the so rare literature about reverse localisation, Schaler [1] defines it as keeping or intentionally introducing linguistic or cultural strangeness into digital content for a particular target locale. The aim is to intentionally differentiate a digital product or service from the dominating culture in that locale [1]. The effect of reverse localisation is manyfold: the product or service in question is certainly set apart from potential competitors, the values and connotations associated with it play on a spirit of adventure, sometimes they just plainly take advantage of existing stereotypes, and almost always cause a sense of curiosity and heightened sense of attention [1].

Reverse localisation differs from normal software localisation in the way of adapting the software. Normal software localisation adapts computer application's linguistic, cultural, regional differences and technical requirements of a source language, culture, region to its target counterparts. The reverse software localisation go from a target language, culture to a source one. As an example, gedit the famous text editor of the Linux environment localised in arabic, will have texts typed from right to left, in arabic characters. Even the systems buttons will be placed at the left instead of the right. If we reverse localise gedit for arabic language, we would introduce to it, some linguistic or cultural features from an underserved culture (for example western date formats) with the aim of intentionally differentiating gedit from the dominating culture in that locale (arabic). The resulting product could have features from both source and target languages.

The recommendation to localisers is to abandon the well-trodden path of localisation as we know it, with all its emphasis on the avoidance of even the smallest sign of cultural diversity and cultural differences [1]. According to the author it will encourage clients to get to know local customs, learn about other languages, taste the delights of foreign cuisine, wear strange clothes, learn how to play the instruments of other cultures; most of all encourage them to bring all their cultural baggage and enjoy the clash of cultures wherever they go and whatever they do.

5 Methodology

Given the advantages of computer applications and ICT tools for digital preservation above, our global approach is to create culturally richer representations like interactive applications based on the cultural values. Cultural differences play a very important role in matching computer interfaces to the expectations of users from different national and cultural backgrounds. But to date, there has been little systematic research as to the extent of such differences, and how to produce software that automatically takes into account these differences [19]. As proposed in [21], the methodology is an adaptation of a 4-phase cross-cultural interface design strategy respectively Investigation, Translation, Implementation and Evaluation.

The Investigation is to learn to understand the processes of the target culture, to Identify its cultural markers which may be included in the interface. A foraging study is an excellent way in which to identify cultural markers, to assess the user attitudes and extract culturally computable features. This phase of investigation is based on common sense knowledge. In this phase, because they are based on common sense of knowledge, some symbolisms or abstractions investigated might be useful or correspond to some global ones. At the same time, some of world known symbolisms may have a wider applicability. So this phase ends with a classification of investigated features.

The Translation phase is to develop a cultural model to enable the developers to determine the similarities and differences present in the targeted culture.

For the Implementation phase, it is to develop a reverse localised application, being sure to employ the culture model established in phase 2 as well as the information obtained in phase 1, and to Implement a prototype and perform usability testing with targeted cultures.

Finally the Evaluation phase help to assess. This user-centered evaluation is an empirical, uses observational evaluation method that ensures usability of interactive systems by including users early and continually throughout user interaction development. The method relies heavily on usage context (for example, user task, user motivation, and so on) as well as a solid understanding of human-computer interaction.

6 Case Study: Preservation of Yemba Culture in a Text Editor

6.1 Investigation of Yemba Cultural Factors

The Yemba culture is rich in traditions that have been preserved and passed from one generation to another since immemorial time. This culture is multifaceted and is expressed in different forms, ranging from its people and language, food, music and dance, art, artifacts, theatre and literature to its ethnic values and ethical norms. It encompasses such things as language, storytelling and oral communication, the use of proverbs as a form of education, art, music, food, spirituality, craftsmanship, history, ancestry, naming, rites of passage, housing, livestock, food, clothing, work, marriage, beliefs, traditional medicine and geography. Some main overt cultural specificities are:

- its language which has 33 character except the X and the Q characters;
- its traditional grassfield clothing consist of skins of either domesticated or wild animals, geometric forms;

– its date and time formats: In Yemba region of Cameroon, every week has eight days and not seven as in the Gregorian calendar. This explains its instability compared to the seven days of the Gregorian week. Each day has a meaning related to activities that are reserved or inspired by the history of the village. Each of these days is related and sometimes suited to particular cultural ceremonies: day market area; day for celebration of traditional ceremonies like traditional weddings and intense agricultural activities days. In general, time is indicated by events that mark: this fact took place during the harvest of particular plants for planting such other dry season or rainy; we say for example "*I was born the year it was the invasion of locusts*"; the Year *where he had eclipse*; "*at sunrise or sunset*". Today, the thing with calendars has evolved in the mother tongue where the days, weeks, months or even years actually have a name.

6.2 Translation of Terms

A Vocabulary of computer terminology and word processing domain in Yemba language was addressed. The localisation of word processing tools for Yemba language and culture demands more than mere translation of computer terms into Yemba. It demanded the creation of Yemba equivalent terms, which involves application of scientific strategies and principles for technical-term creation done with the help of Yemba study committee.

6.3 Computational Results

To integrate Yemba cultural factors into the localised text editor environment, we need to tackle the following issues: colour schemes, pictures and images, sounds, historical data, hand signals, symbols, product names and acronyms. But to the far of our knowledge, these cultural features are not formalized at all, except the date, time and days representation issues. The Yemba calendar has 12 months. Each week has eight days, instead of 7 days in the generic calendar. The days are Mbouowa, Mbouolo, Meta, Mbouokeu, Mbouotchou, Efaa, Djielah and Ngan. As they are not pair with the normal days of a week, we cant associate them with. The work of reverse localisation which is of national scale, offers a culturally-based framework to process words.

The prototype of our reversed localised text editor shown in Fig. 1. is built using the architectural pattern Model-view-controller (MVC). The model is represented by the domain-specific representation of the information on which the application operates, i.e. text processing. The View renders the model into a form suitable for interaction like in Fig. 1 The Controller is composed of processes and responds to events, typically user actions, and invokes changes on the model and perhaps the view. The modeling of the text editor application, is done by using use UML language and the resulting protype is developed with java programming language. It supports the following basic features such as Insert and Delete text, Copy, Cut and Paste, Page size and margins, Search and replace, Word wrap, Print. Our reverse localised text editor supports additional features

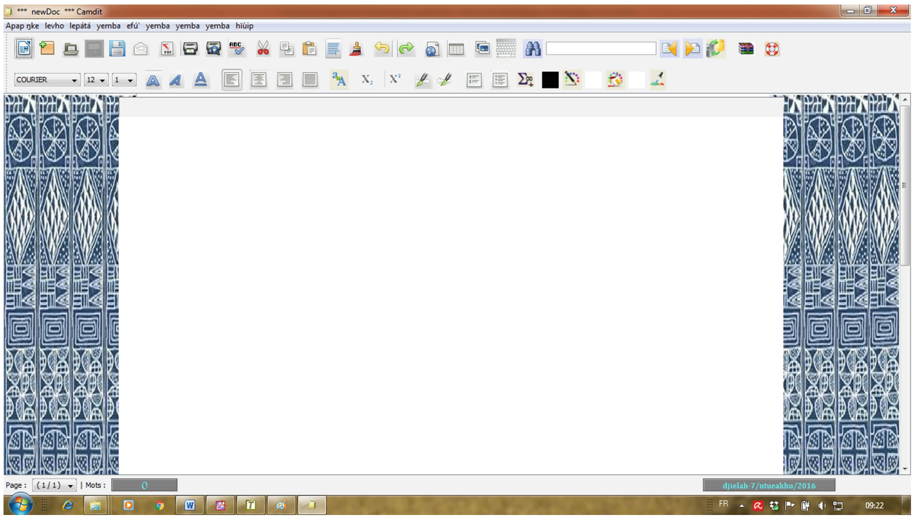

Fig. 1. Prototype of a reverse localised text editor in Yemba culture

that enable a user to manipulate and format documents in more sophisticated ways. These full features are Management files capabilities, font specifications, Spell checker, WYSIWYG (what you see is what you get) functions.

6.4 Issues and Trade-offs

The reverse localisation results in solving many issues from linguistic to hardware and software issues. For the integration of Yemba main cultural factors in the localised text editor environment, we identified some of the problems and addressed them.

- **Terminological Issues:** Language problems are often caused by terminology. Whenever English language software is translated into a local language, decisions are taken on mapping from English terms to local terms. Inevitably, some measures of arbitrariness are 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 towards 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.
- **Linguistic Issues:** The development of the Yemba language in the direction of technological development has suffered over the years due to the use of English as the language of technology in Cameroon. Hence, many of the English terms used in nowadays text editors do not have corresponding Yemba terms. Therefore it was necessary to develop terms that can convey the meanings of the original English terms to the Yemba user.

- **Hardware Issues:** The localised text editor must support the character set of the Yemba language and must be configured to present numbers and other values in the local format.
- **Software Issues:** Localising text editor might require adding other software such as a new spell checker that recognizes words in the local language. More over, the new localised tool will need a tool for statistical purposes, grammatical corrections, search tools, ... The creation and translation of localised content is more demanding, mostly in respect of language skills and awareness of tools appropriate to the work, such as appropriate fonts and the means to code in extended Latin and non-Latin scripts.
- **Shortcut Issues:** There are varying degrees of reverse localisation. Yet there are no obvious criteria for guiding the appropriate level of localisation. Evidence from our study of localised text editors 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 [22]. When the well known shortcut keys are mapped to a localised Yemba version of the File menu, some of these mnemonics are inappropriate. For example, because of the inexistence of the X character in Yemba language, the shortcut Crtl-X helping to cut texts is not suited.

All these issues results in trade-offs between performance, the ease of use, and the ease of implementation.

7 Conclusion

Our attempt to the cultural reverse localisation of an text processing system in the Yemba culture, can contribute to fill the gap between computer programs designers, developers, and final users whose culture is Yemba. Our localised text editor offers to Yemba speakers a tool to easily produce documents in their language, because a language is the mirror, the soul of the people and its culture, its mark of existence. Our work, extensible to other languages, can contribute to the preservation of underserved cultures in computer applications. This proposed research aims at investigating the need for a cultural interface that extends beyond the traditional translated interface. By making use of cultural models and exhaustive usability testing, the most effective culturally sensitive interface can be discovered and recommended for use with a text editor application. When fully implemented, the proposed system will contribute to lessen the cultural digital divide observed in computer applications.

Acknowledgments. We thank the Yemba language Study Committee, the anonymous reviewers for their careful reading of our manuscript, for their detailed and constructive comments and suggestions.

References

1. Schäler, R.: Reverse Localisation. Localis. Focus Int. J. Localis. **6**(1), 39–48 (2007)
2. Kersten, G.E., Matwin, S., Noronha, S.J., Kersten, M.A.: The software for cultures and the cultures in software. In: Hansen, H.R., Bichler, M., Harald, H. (eds.) Proceedings of the 8th ECIS 2000, Vienna, Austria, vol. 1, pp. 509–514. (2000)
3. Wolff, F.: Effecting change through localization: localization guide for free and open source. IDRC, Canada (2011)
4. Tedre, M., Kähkönen, E., Kommers, P.: Is universal usability universal only to us? In: Proceedings of ACM Conference CUU 2003, Vancouver, Canada (2003)
5. Reinecke, K., Schenkel, S., Bernstein, A.: Modeling a users culture. In: Blanchard, E.G., Allard, D. (eds.) The Handbook of Research in Culturally-Aware Information Technology: Perspectives and Models. IGI Global, Hershey (2010)
6. Yacob, D.: Localize or be localized: an assessment of localisation frameworks. In: The International Symposium on ICT: Education and Application in Developing Countries. IEEE Press, Addis Abeba (2004)
7. Keller, B., Perez-Quinones, M., Vatrapu, R.: Cultural issues and opportunities in computing education. In: 9th International Conference on Engineering Education, pp. 14–18 (2006)
8. Reinecke, K., Reif, G., Bernstein, A.: Cultural user modeling with CUMO: an approach to overcome the personalization bootstrapping problem. In: Proceedings of the First International Workshop on Cultural Heritage on the Semantic Web at the 6th ISWC, pp. 83–90 (2007)
9. Hofstede, G.: Cultures and Organizations: Software of the Mind. McGraw-Hill Companies, Inc., New York (1997)
10. Hestres, L.E.: The influence of American culture on software design: Microsoft Outlook as a case study. GNOVIS Georget. Univ. J. Commun. Cult. Technol. **3**(1) (2003). 22 pages
11. Beelders, T., McDonald, T., Blignaut, P.: A proposed study to determine the effect of culture on the usability of word processors. In: Proceedings of CHI-SA 2005, pp. 29–33 (2005)
12. Tedre, M., Sutinen, E., Kahkonen, E.: Ethnocomputing: ICT in cultural and social context. Commun. ACM **49**(1), 126–130 (2006)
13. Bartolotta, M., Di Naro, S., Brutto, M., Misuraca, P., Villa, B.: Information Systems for preservation of cultural heritage, In: International Archives of Photogrammetry and Remote Sensing, XXXIII, Amsterdam, pp. 864–871 (2000)
14. Internet World Stats. http://www.internetworldstats.com/stats7.htm. Accessed 9 Mar 2017
15. Cultural Politics. http://culturalpolitics.net/digital-cultures/cultural-divide. Accessed 19 Mar 2017
16. Vilbrandt, C., Pasko, G., Pasko, A., Fayolle, P.A., Vilbrandt, T., Goodwin, J.R., Goodwin, J.M., Kunii, T.L.: Cultural heritage preservation using constructive shape modeling. Comput. Graph. Forum **23**(1), 25–41 (2004)
17. Rauterberg, M.: From personal to cultural computing: how to assess a cultural experience. In: Kemper, G., von Hellberg, P. (eds.) uDayIVInformation nutzbar machen, pp. 13–21. Pabst Science Publ. (2006)
18. Michael, K., Dunn, L.: The use of information and communication technology for the preservation of Aboriginal culture: the Badimaya people of Western Australia. In: Dyson, L. (ed.) Information Technology and Indigenous People. Idea Group Publishing, Hershey (2006)

19. Anacleto, J., Lieberman, H., Tsutsumi, M., Neris, V., Carvalho, A., Espinosa, J., Godoi, M., Zem-Mascarenhas, S.: Can common sense uncover cultural differences in computer applications? In: Bramer, M. (ed.) IFIP AI 2006. IIFIP, vol. 217, pp. 1–10. Springer, Boston (2006). https://doi.org/10.1007/978-0-387-34747-9_1
20. Smith, A., Reitsma, L., Van den Hoven, E., Kotzé, P., Coetzee, L.: Towards preserving indigenous oral stories using tangible objects, In: Proceedings of the 2nd ICCC, Amsterdam, pp. 86–91 (2011)
21. Jagne, J., Smith, S.G., Duncker, E., Curzon, P.: Cross-cultural interface design strategy. Technical report, Interaction design Centre, Middlesex University (2004)
22. Grigas, G., Jevsikova, T., Strelkauskyt, A., Cox, S.: Localisation issues of software shortcuts. Int. J. Localis. **11**, 40–53 (2011)