Variable Menus for the Local Adaptation of Graphical User Interfaces

An Implementation Approach

Cristina Olaverri-Monreal [a,d]
[a] Instituto de Telecomunicações, DCC, Faculdade de Ciências da Universidade do Porto, Rua do Campo Alegre 1021/1055, 4169-007, Porto, Portugal.
Email: cristina.olaverri@dcc.fc.u.pt

Christoph Draxler [b]
[b] Institut für Phonetik und Sprachverarbeitung Ludwig-Maximilians-Universität Schellingstr. 3, D-80799 Munich
Email: draxler@phonetik.uni-muenchen.de

Klaus-Josef Bengler [c,d]
[c] Faculty of Mechanical Engineering Technische Universität München D-85748 Garching, Germany
[d] BMW Forschung & Technik Hanauerstr. 46, D-80992 Munich
Email: bengler@lfe.mw.tum.de

Abstract—**Interaction with an insufficiently localized Graphical User Interface can lead to a decrease in the ease of navigation through the system. Poor interfaces entail frustration for the user and reduce the system's usefulness. Thus all elements of Graphical User Interfaces should be subjected to localization and a variable menu structure should be part of the goals of software localization efforts. Considering the extensive use of XML in the implementation of hierarchical menu structures, we propose to use an implementation approach for the local adaptation of menus that is equivalent to the procedure used to create an XLIFF file. The locally adapted menu structure can then be merged with the translated content into a target localized XML document.**

Keywords: XML, XSLT, XLIFF, localization, internationalization, menu structure

I. INTRODUCTION

The worldwide expansion of markets is one of the most important characteristics of the economical development. Above all when through the use of the World Wide Web, companies have the possibility to expand globally [1, 2, 3, 4]. This development implies the adaptation of their products to different markets: when introducing a product into a certain target market the user should be able to interact with it in his mother tongue and get the impression that the particular product was developed exclusively for his country [5]. This fact together with the increasing markets' globalization is leading to an increase of the significance of a linguistic, cultural and legal adaptation of a specific product to different countries (Localization) as well as the consideration of this adaptation early in the development phase of the product (Internationalization). In the course of the worldwide expansion of the markets, several studies have investigated the role of cultural diversity regarding the use of products concluding that interaction with a Human Machine Interface depends on the target culture [6], [7]. In the case of systems that are to be operated by users with different cultural background it is a common practice to evaluate the existing differences in a variety of cultures [8].

Currently, most of the graphical user interfaces (GUIs) are organized in the form of hierarchical menus. The fluency of communication between the user and the system is determined by the kind of navigation among the different options from the graphical user interface. Consequently, the organization of the menus inside an application determines a smooth interaction with the system.

Development of a user interface without proper consideration of component integration can increase the cognitive load, errors and annoyance for users [9]. Thus, ease of navigation through the menu improves the user experience with a certain product and the process of learning its use in contrast to the frustration that can originate a poorly designed user interface**.** The localization of menu structures can provide increased user friendliness [10]. Therefore, first priority when trying to increase the user experience is to organize the menus in the display in such a way that they facilitate the interaction between the user and the system. For example, the use of a certain option in a document preparation system can be widely spread in a specific country while in another the same option can be seldom used. Positioning this function prominently in the menu structure facilitates the menu navigation to this frequently used function. At the same time locating the same option in a less emphasized position when it is not frequently used assures a smooth navigation and interaction with the system. Currently only terminology aspects are considered when localizing user interfaces. An adaptation of the menu structure to the target culture is, in general, not taken into account.

The authors in [11] suggested an implementation approach that facilitates the adaptation of menu structures in Driver Information Systems using XML related technologies and accessing an independent vocabulary file. Relying on this approach and since XLIFF supports the elements for a variable menu structure that should be part of the goals of software localization efforts; we present in this paper an implementation approach for the local adaptation of menus that is equivalent to the procedure used to create an XLIFF file.

A. *Internationalization*

The cost of software localization can be strongly reduced by taking into account different rules in the previous internationalization process of a software product during its development [12, 13]. After internationalization, cultural adjustments can be made with relatively slight changes, while the localization of a finished product that has not been internationalized often requires a comprehensive creation of new different versions. The development of internationalized software follows several requirements that can be summarized as the separation of source code and elements that are to be altered during the localization process. These requirements for producing internationalized software applications were formulated by Esselink as follows [12]:

- The implementation methodology of an internationalized system must enable a fast and simple transfer into a localized system.
- In order to be able to display characters from various types of language systems, Unicode must be used.
- The structure of the menu must be stored separately from the language. This allows translating the text without modifying the document structure achieving therefore a dialog control that is independent from the respective language.
- Character size and field length should not be specified. The size of the texts and/or buttons should be adapted automatically to the user interface.
- For all elements of the user interface separate source files should be created. Also the diagrams and images should be stored separately.

B. *Related Technology*

One of the difficulties in the software localization process is the use of different platforms, data formats and character sets of non English software product versions. Sometimes an application that was designed in English can only be localized on another platform, version or environment in a different language. The diversity of these software platforms and technologies are reflected in the variety of tools that support the localization process and that are frequently incompatible with each other [14]. Thus, there is an urgent need for a language that assures a platform independency and ensures to save documents and data in a structured form.

XML, the Extensible Mark-up Language, meets all these requirements and enables a smooth document exchange. Furthermore, a complete multilingual compatibility is guaranteed through the Unicode default character encoding [15]. XML permits the creation of structured documents since it identifies the text parts of a document using elements ordered in a hierarchical form. This characterization makes it possible to represent each element's information in a graphical tree structure [16].

The XML-based XSL (Extensible Style sheet Language) is tree-oriented and is used to format XML documents independently of the content [17]; it is comparable to the Cascading Style Sheets (CSS) for HTML [18]. The XSL component XSLT (Extensible Style sheet Language Transformations) allows the manipulation and transformation of an XML document into a different XML having previously defined the rules for the transformation process [19, 20]. In the same way that XSLT is used for XSL Transformation of documents described in XML, it is also used to control the machine translation of XML documents such as XLIFF [21].

XLIFF is the XML Localization Interchange File Format developed by the Organization for the Advancement of Structured Information Standards (OASIS) [22]. XLIFF is based on TMX 1.2 open standard and the XML-based format for localizable content Open Tag version 1.2. TMX 1.2 adopted XML for the Translation Memories (TM) storage and exchange. Thus, XLIFF is compatible with existing XML localization industry standards vocabularies such as TMX for Translation Memory Exchange (TMX) and Term Base Exchange (TMX) [14]. Like the other mark-up languages it also represents menu hierarchies including marking signs for data description [23, 24, 25]. These marking signs can be used to define options and sub options hierarchically. XLIFF simplifies the localization process due to the fact that its structure includes a single source language and a single target language. Fig.1 shows a XLIFF sample code for describing one of its trans-unit elements. This simplicity avoids having different versions of multilingual files that lead to an unclear and confusing localization phases: In addition, CAT tools build around the XLIFF standard. The current resources within XLIFF 1.2 have extensive resources for dealing with inline elements that are important to translation. Moreover it can represent localizable menu structures.

II. XLIFF AS A METHOD TO LOCALIZE VARIABLE MENU STRUCTURES

XLIFF was conceived as a single interchange file format for software localization through XML vocabularies and it aims to be an interchange data format for the complete localization procedure. XLIFF provides useful resources for the software localization as an intermediary file format to translate multilingual documents [26]. In addition, since XLIFF contains elements for the description of the tree level documents, it seems ideal to be used as a format to represent localizable menu structures. Our approach encourages extending XLIFF in some worthwhile aspects that address the localization of menu structures: In the same way that the content of the source XML file is extracted into an XLIFF file, we propose accessing the tree nodes of the basic XML document using the template rules of an XSLT Style sheet to transform a source XML document into a XLIFF document with a local menu structure. Thus, the locally adapted menu structure can be merged with the translated content into a target localized XML document. Translatable text strings are extracted from an XML file into a XLIFF document applying XSLT Style sheets creating thus a new XML-based vocabulary file. Not only linguistic components are to be considered as localizable resources but also menu structures need to be adapted to a particular culture. Thus, if the XLIFF file contains not only text data but also a localized menu structure for every locale, an efficient translation of the menu can be made that already follows the appropriate dialog structure for a particular target culture.

```
<trans-unit>
<source xml:lang="en-US">Hello
world!</source>
<target state="translated" xml:lang="es-
ES">¡Hola mundo!</target>
</trans-unit>
```

Figure 1. XLIFF sample code describing a trans-unit element

XSL can be used to transform an XML file directly into an XLIFF file with the cultural adapted menu structure. The text in the XLIFF file will then be translated and merged back into the original XML format file. Fig. 2 illustrates the process of the menu structure transformation through XSLT template rules. These templates enable the extraction and reintegration of linguistic components in the process of creating a XLIFF file. It seems thus, a logical consequence to make the most of this action accessing not only the nodes that identify translatable text strings but also the nodes representing the menu of a particular application.

The XLIFF architecture considers the possibility of including attributes in both the source and the target elements. Thus, the menu structure can easily be defined within the <context-group> element that allows storing resource metadata related to the trans-unit. Currently the <context-group> named group element provides the regularity and specificity necessary to capture menu structures since it contains context elements corresponding to the level in the tree in which it occurs. It can be used for example to specify if a particular <context-group> element should be displayed or not. Thus, it describes the necessary resources to accomplish the localization of menu structures. As a consequence, the sub-language of elements in XLIFF relating to inline phenomena (<g>, <x/>, <bx/>, <ex/>, <bpt>, <ept>, <sub>, <it>, <ph>) offers the perfect frame for a menu structure description. In addition, the structural <group> element specifies a set of elements for being processed together. Since a <group> element can contain other <group> element it can be used to describe the hierarchy of a dialog based on menus. The subordinated to the group element attributes menu, menu-name and menu-option which describe the resources contained within the <group> provide the ideal environment to describe the culture based variation in menu structures.

Fig. 3 shows an example of a partial menu structure defined with XLIFF for the particular case study of a Driver Information System.

III. IMPLEMENTATION

In this paper we present an implementation approach that facilitates the adaptation of menu structures to local cultures. To determine the options that should be included in the different local versions a previous international study of user preferences, regarding the location of menu options in a GUI, should be performed. We describe in this paper a menu structure that, according to internationalization requirements, is independent of the content. This independency is achieved by the use of XML and XSLT and consequently XLIFF.

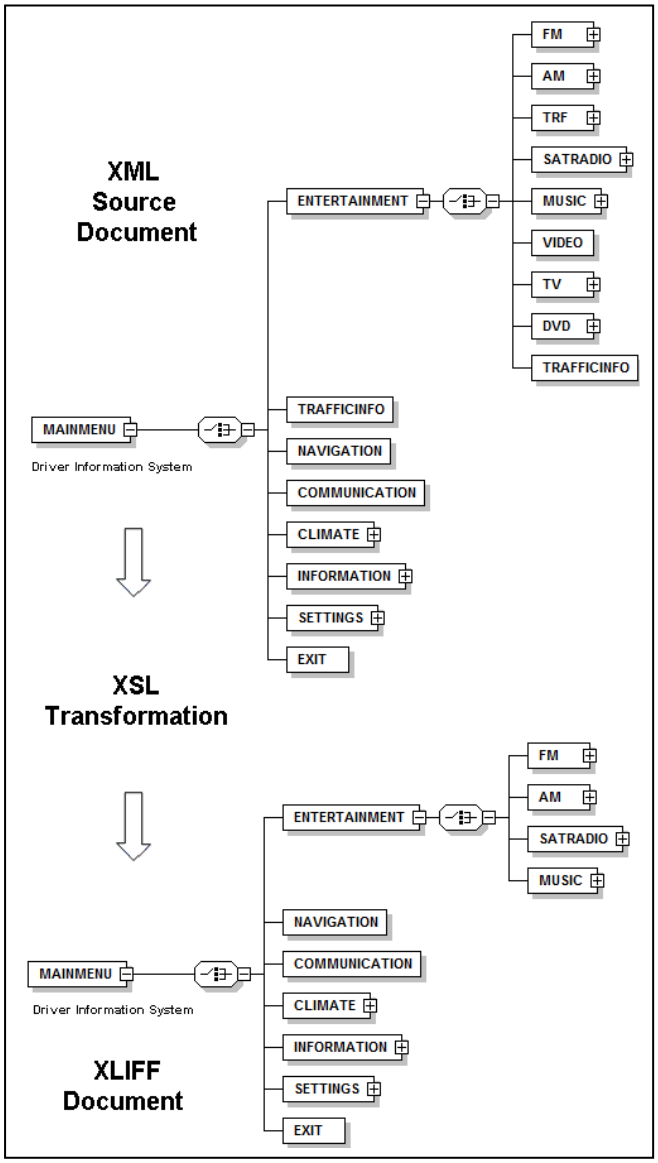

Figure 2. XML source document menu structure transformation into XLIFF target document. The tree nodes of the XML document are selected using the template rules of an XSLT Style sheet to transform a source XML document into a XLIFF document with a local menu structure.

The use of locale standards (character set, date and time format) and a particular language for a culture are built into each XSLT, the document that generates the representation for the respective country. In addition the elements can be organized in a certain order using the "sort" function and font size and field length can be programmed by a function, remaining variable. This is particularly useful and necessary for localization into other writing systems. Moreover it assures the system internationalization [12].

We build separate source files for all elements of the user interface to achieve independency between the texts of the individual options and the layout. With this method only minimal changes to the code are necessary in order to accomplish a fast localization of the menu.

```
<?xml version="1.0" encoding="UTF-8"?>
<xliff version="1.2">
 <file original="DISsample.xml" source-
language="en-US">
   <header>
    <skl>
      <external-file href="DISsample.skl"/>
    </skl>
   </header>
    <body>
      <group id="DISmainmenu" restype="menu"
resname="sid mainmenu">
       <trans-unit id="1" restype="menuitem"
resname="sid_entertainment">
         <source xml:lang="en-
US">Entertainment</source>
         <target state="translated"
xml:lang="de-DE">Unterhaltung</target>
       </trans-unit>
       <trans-unit id="2" restype="menuitem"
resname="sid navigation">
         <source xml:lang="en-
US">Navigation</source>
         <target state="translated"
xml:lang="de-DE">Navigation</target>
       </trans-unit>
      </group>
   </body>
 </file>
</xliff>
```

Figure 3. Sample XLIFF code describing a partial Driver Information System's menu structure. The code includes the text strings for the source language English and target language German.

Transforming a common menu structure for all variants, we produce a localized menu for a certain locale. The rules to define the different positions of the options within the menu are included in the template and enable to create a XLIFF document with extracted localizable text elements, extracted additional localizable content and a localized menu structure.

A. XLIFF document generation

A menu structure, with all possible options' variants is designed in XML. This structure is modified through selection of relevant nodes and transformed into the respective local versions. The elements "main menu", "menu" and "menu item" describe the tree structure in all of the XML documents according to the appropriate XML Schema definition. The "Root" element or point of origin of the tree structure is represented by "main menu". All elements of lower hierarchies are designated with the name "menu" or "menu item". The element "menu item" describes a terminal tree node in a hierarchy level, and options where a function is executed.

The transformation into local structures is implemented by accessing the tree nodes of the basic XML document, namely the elements with the names "menu" and/or "menu item" using an XSLT Style sheet. Options from the basic menu are located in different positions depending on the target culture in the localized versions. The variable options are identified through pattern matching. Afterwards, the content of the XML document is selectively copied into a new XLIFF document which represents the localized structure. XSLT allows not only copying certain nodes but also sub trees that in our implementation correspond to "menu" and "menu item" elements inside a "menu" element. This is applied in the structure parts that are common for both, the basic XML and the localized version. In the same process the nodes representing the text strings are extracted for each locale.

B. Generation of the localized document

The independence of content and menu structure is achieved in XLIFF through the extraction of the text strings to be translated. In addition to these strings, XLIFF includes the correspondent elements' attributes for the menu structure definition. After the translation of the extracted text and the localization of the extracted additional content, the document is merged into the final localized XML document containing the respective text of both, the particular language and the menu structure of the culture. Fig. 4 illustrates the transformation process of a source XML document describing a menu.

In the source XML documents, the nodes corresponding to the menu options and to the text elements to be extracted into the XLIFF file are defined through ID attributes. The XLIFF file, in its turn, includes the same attributes with the correspondent string translation.

A fusion of both the XLIFF file with the target XML file is achieved through comparison of the ID attributes of both documents and latter insertion of the correspondent menu structure and translated text into the target XML document.

In this way a locale-specific resource file for the current user's locale can be loaded; in this case, the XLIFF files containing the translated texts.

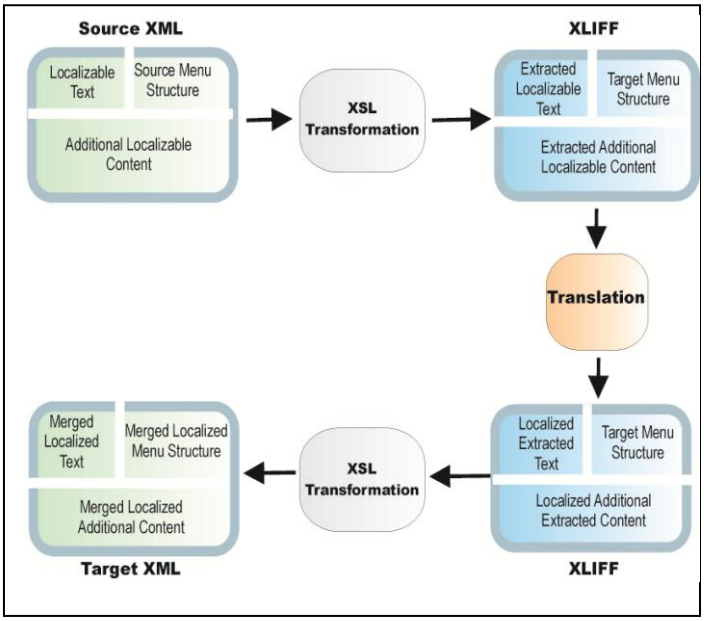

Figure 4. XML menu localization process via XLIFF. The content of the source XML file is extracted into an XLIFF file together with a local menu structure. After translation of the extracted localizable text and localization of the additional extracted content, the locally adapted menu structure is merged into a target localized XML document.

IV. CONCLUSIONS

A cross cultural survey related to the interaction preferences with Driver Information Systems showed how interaction with such systems can vary depending on the target culture and builds the foundation for a need in the cultural adaptation of menu structures [27]. Separation of variable elements and code is traditionally achieved by storing layout elements in different files. The storage of locale-specific information in resource bundles enables a program code that is independent from the user's locale.

We use XML to represent a common menu structure, which can be subsequently and independently transformed into XLIFF documents using XSLT templates.

The initial program structure enables us to add variable parameters from separate files [28, 29]. This can represent a special challenge in software with a multilingual user interface, since the international basis can concern also the structure of an object [30].

In our approach this structure is altered in selected nodes, in order to produce different versions. We achieve not only an independency of both, menu content from structure, but we also enable varying the structure itself without having to implement a complete system for each variant. The main advantage of this approach is that the basic structure is extendible and allows the modification of its elements (i.e. both the position and the number of nodes.) Thus, an application can be developed without requiring the full information about the final elements' positions because modifications can be easily implemented thereafter. For instance, we can create a unique XML structure that can be modified to be tested in usability tests for different cultures. In addition, depending on the development phase of the application, new functions that are going to be considered for latter versions can be included in the system.

XLIFF, due to its characteristics and since it aims to be an interchange data format for the whole localization process, constitutes the perfect framework to implement a basic expandable structure, since it facilitates the insertion and adaptation of new elements, in this case, menu options. However, if higher programming languages need to be used to implement an application, a more time consuming complete system for each variant has to be designed.

Complying with internationalization standards implies using a special solution through variants. We consider variants already in the development phase of a product storing the applications' data within many unique resource files that will be processed in the localization procedure. Thus, by the use of XML-based languages like XLIFF, XML and XSLT we assure not only a platform independency but also a structured data representation and multilingual consistency.

In cases where the interaction with a certain software application demands a considerable amount of cognitive processing tasks, like for example in a vehicular environment, our approach can avoid a delay or worse, an interruption of the cognitive processing of road-related information preventing thus longer reaction times.

Therefore, in-vehicle infotainment systems represent a particular well suitable framework to apply our implementation approach. The cultural adaptation might enable the driver to focus on the decision-making process; awareness and diagnosis of road situation, minimizing thus the distraction factors that may interfere with the primary duty of driving and jeopardize road safety.

REFERENCES

[1] T. Welzer, D. Riaño, B. Brumen, and M. Družovec, "Internationalization Content in Intelligent Systems – How to Teach it" M.Gh. Negoita et al. (Eds.): KES 2004, LNAI 3214, pp. 1039–1044, 2004. Springer-Verlag Berlin Heidelberg.

[2] O. De Troyer, and S. Casteleyn, "Designing localized web sites", Web Information Systems – WISE, 2004, Proceedings Lecture Notes in Computer Science 3306: 547-558.

[3] P. Norris, "Digital Dvide: Civic Engagement, Information Poverty and the Internet World Wide" Cambridge University Press, 2001.

[4] J. De la Torre, and R.W. Moxon, "Introduction to the Symposium E-commerce and Global Business: The impact of the information and communication technology revolution on the conduct of international business" Journal of international business studies 32 (4): 617-639, 2001.

[5] The Localization Industry Standards Association (LISA). Available: www.lisa.org [Accessed 19 January 2011].

[6] D. Ford, C. Connelly, and D. Meister, "Information systems research and Hofstede's culture's consequences: an uneasy and incomplete partnership," Engineering Management, IEEE Transactions on, vol. 50, no. 1, pp. 8–25, 2003.

[7] R. Heimgärtner, "Cultural Differences in Human Computer Interaction: Results from Two Online Surveys", Open Innovation: Neue Perspektiven im Kontext von Information und Wissen, p. 145, 2007.

[8] P. Bourges- Waldegg, and S. Scrivener, "Meaning, the central issue in cross-cultural HCI design" Interacting with Computers 9 (3): 287-309,1998.

[9] J. Lee, and B. Kantowitz, "Network analysis of information flows to integrate in-vehicle information systems". International Journal of Vehicle Information and Communication Systems, 1, 24-43, 2005.

[10] R. Heimgärtner, "Towards Cultural Adaptability in Driver Information and -Assistance Systems". Usability and Internationalization. Global and Local User Interfaces, 2007. Beijing, China: Springer.

[11] C. Olaverri Monreal, M. Breisinger, K. Bengler and C. Draxler, "Markup Languages and Menu Structure Transformation during the Internationalisation process of Driver Information Systems"; Localisation Focus, vol. 9, pp. 4–12, 2010.

[12] B. Esselink, "A practical guide to localization", John Benjamins, 2000.

[13] D. Tuffley, "Optimizing a Software Internationalization Curriculum. Designing for Global Markets 4" – Proceedings of the IWIPS 2002, July 11th-13th, 2002, Austin Texas.

[14] P. Reynolds, and T. Jewtushenko, "What Is XLIFF and Why Should I Use It? :A brief overview of the XML Localisation Interchange File Format (XLIFF)", 2005, XML Journal [Online] Available: http://xml.sys-con.com/node/121957 [Accessed May 2010].

[15] M. Bradin, "XML Improves Localisation Projects". Multilingual Computing &Technology 1, 2, 2002.

[16] H. Lobin, "'Informationsmodellierung in XML und SGML", Berlin, Heidelberg: Springer- Verlag, 2001.

[17] M. Van Otegem, "Sams teach yourself XSLT in 21 days". European Journal of Control, 15, 2002.

[18] Y. Savourel, "XML Technologies and the Localisation Process". MultiLingual Computing & Technology Nr. 35 11 (7), 2002.

[19] Y. Yu, J. Lu, J. Mylopoulos, W. Sun, J.H. Xue, and E.H. D'Hollander, "Making XML document markup international", Software: Practice and Experience 35 (1), 1-14, 2005.

[20] S. Adler, A. Milowski, J. Richman, and S. Zilles, "Extensible Stylesheet Language (XSL)-Version 1.0", 2001.

[21] P. Senellart, and J. Senellart, "SYSTRAN translation stylesheets: machine translation driven by XSLT". 2005.

[22] OASIS XLIFF Version 1.2 (http://docs.oasis-open.org/xliff/v1.2/os/xliff-core.html#context-group). [Accessed May 2010].

[23] R. Eckstein, and M. Casabianca, "XML: Kurz& gut", O'Reilly Germany, 2001.

[24] M. Sperberg-McQueen, C. Huitfeld, and A. Renear, "Meaning and interpretation of markup". Markup Languages 2/3, 215-234, 2000.

[25] P. Whitehead, E. Friedman-Hill, and E. Vander Veer, "Java and XML: your visual blueprint for creating Java-enhanced Web programs", Third Avenue New York, NY 10022: Wiley Publishing, Inc. 909, 2002.

[26] R.M. Raya, "XML Localisation Interchange File Format as an intermediate file format" [Online] Available: http://www.maxprograms.com/articles/xliff.html [Accessed Jun 2010].

[27] C. Olaverri Monreal and K.J. Bengler "Impact of cultural diversity on the menu structure design of Driver Information Systems: a cross-cultural study", unpublished.

[28] P.A.V. Hall, and R. Hudson, "Software without Frontiers - A Multi-platform, Multi-cultural, Multination Approach", Wiley, New York, 1997.

[29] P. Rößger, "Cross Cultural Differences in Human Machine Interfaces of Driver-Information-Systems and how to Cope with them from the Software Side." Paper presented on the ITSA Meeting, 2003, Minneapolis, MN.

[30] E.A. Karkaletsis, C.D. Spyropoulos, and G.A. Vouros, "A knowledge-based methodology for supporting multilingual and user-tailored interfaces" , Interacting with Computers 9 (3): 311-333, 1998.