



University of Fort Hare
Together in Excellence

Cultural and Linguistic Localization of the Virtual Shop-Owner Interfaces of E- Commerce Platforms for Rural Development

A thesis submitted in fulfillment of the requirements of the degree of

Master of Science

in

Computer Science

at the

University of Fort Hare

by

Siyabulela Dyakalashe

November 2009

Declaration

I hereby confirm that the content presented in this research study is my own original work and information extracted from other external sources has been correctly acknowledged and referenced.

.....

Siyabulela Dyakalashe

November 2009

Acknowledgements

It has been an honour to work alongside such great people with great minds during the period of completing this M.Sc exploration. I would like to use this opportunity to thank all those who guided, supported and encouraged me in successfully completing this research.

First of all, I would like to thank Our Lord for guiding and helping me through the difficulties and obstacles encountered during the research.

Secondly, I would like to thank my Family for being supportive of my ambitions in life, both academically and socially. *May the Lord, Our Father, bless you all.* I would also like to dedicate this research to my late older sister, Ncebakazi Dyakalashe, who was so inspirational of my academic carrier. Your beautiful memories are engraved in our hearts and minds.

I would like to thank Prof. J. Chadwick and my supervisors, Prof. H.N. Muyingi and Prof. A. Terzoli, of the Computer Science Department for the intelligent ideas, comments and suggestions which lead to the development of this novel piece of research. I am grateful for your guidance in helping me broaden my horizon and for grooming my mind for future challenges. Your support is much appreciated.

I would also like to thank M. Thinyane and S.G. Njeje for assisting me during the development and compilation of the software discussed in this research. The encouragement and support you have both given me has not gone unnoticed.

I would like to acknowledge the excellent research collaboration between the University of Fort Hare and Rhodes University students in the development of rural and marginalized communities in the Eastern Cape. Thanks to the University of Fort Hare CoE sponsors, THRIP, Telkom, Tellabs, Grintek, Amatole Telecommunication Services, Mars Technologies, and DRISA for the research funding and support in successfully completing the research.

Last but not least, I would like to thank my colleagues and friends for their support during these past two years of completing my research. I wish you all the best in the future and for those in

the process of completing their M.Sc, I wish you all the best in successfully carrying out your research.

Abstract

The introduction of Information and Communication Technologies (ICTs) for rural development in rural marginalized societies is vastly growing. However, the success of developing and deploying ICT related services is still in question as influential factors such as adaptability, scalability, sustainability, and usability have great effect on the rate of growth of ICTs in rural environments. The problem is that these ICT services should be maintained and sustained by the targeted communities. The main cause for rural marginalization is the fact that some communities situated in rural settings are educationally challenged and computer illiterate or semiliterate in comparison with urban communities.

An ICT for development (ICT4D) intervention in the form of an e-Commerce platform that targets the social and economic growth of rural marginalized communities has been developed and field tested at Dwesa, a rural community located on the Wild Coast of the former homeland of Transkei in the Eastern Cape Province. The e-Commerce platform is known as “*buy at Dwesa*” and can be visited at this URL, <http://www.dwesa.com>. The aim of the e-Commerce platform is to motivate small entrepreneurs in rural areas to market their products and themselves to the global market as they lack the skills and resources for marketing their art and crafts. Virtual stores are created for a small group of entrepreneurs who will maintain and sustain the stores on their own. These entrepreneurs are often elderly women with limited education and little to no computer literacy - meaning that sustaining the stores may prove difficult for them.

In this research we discuss the re-design and re-development of the virtual shop-owner interfaces of the e-Commerce platform to make them more culturally and linguistically localized. The virtual shops allow shop-owners to upload their artifacts to advertise and sell on the customer’s end of the e-Commerce platform. For multilingual and multicultural communities, adoption of the software interfaces to the user’s cultural and linguistic needs and modes of expression is important as failure to do so may reduce the level of benefits of e-Commerce initiatives.

Table of Contents

CHAPTER ONE: Introduction	1
1.1 Introduction	2
1.2 Research Problem Statement	2
1.3 Research Objectives.....	4
1.4 Research Context	6
1.5 Thesis Outline.....	8
 CHAPTER TWO: Literature Review and Related Work.....	 9
2.1 Introduction	10
2.2 ICT and Localization for Rural Development.....	11
2.3 Dwesa Shopping Mall.....	13
2.4 Culture and Usability.....	15
2.5 Localization	17
2.5.1 Cultural Localization.....	17
2.5.2 Linguistic Localization	18
2.5.3 Audio in Localization	19
2.6 Localization and Translation Tools.....	20
2.6.1 Translation Memory and Machine Translation.....	21
2.6.2 Translation Databases and Portals.....	22
2.7 Conclusion.....	26
 CHAPTER THREE: Research Methodology and System Design	 27
3.1 Introduction	28
3.2 Methodology.....	28
3.2.1 Throwaway Prototyping.....	30
3.2.2 Evolutionary Prototyping	31
3.3 Dwesa ICT4D Infrastructure and Services Overview.....	33
3.3.1 Dwesa Shopping e-Mall Background	34

3.4	System Architecture.....	37
3.4.1	User Roles	37
3.4.2	LAMP Architecture	38
3.5	Shop-Owners' End.....	39
3.5.1	Viewing Uploaded Shop Items	40
3.5.2	Uploading Shop Items	45
3.5.3	Viewing Customer Orders	46
3.5.4	Navigation Support Tool	48
3.5.5	Language Localization	49
3.5.6	Color Picker	52
3.6	Audio Clues	54
3.7	Conclusion.....	54
CHAPTER FOUR: Implementation.....		55
4.1	Introduction	56
4.2	Database Connection	56
4.3	Preloading User Settings.....	57
4.4	Language Selection	59
4.5	Background Colour Selection	63
4.6	Image Gallery and Slideshow	65
4.6.1	Image Slideshow	65
4.6.2	Image Gallery	67
4.7	Autocompletion and Autosuggestion	69
4.8	Drag and Drop Sortables.....	70
4.9	Audio Interaction	74
4.10	Administrators' Help	75
4.11	Conclusion.....	76
CHAPTER FIVE: Analysis and Findings		77
5.1	Introduction	78
5.2	System Evaluation	78
5.2.1	Software Designers' Perspective	78

5.2.2	Participants' Perspective.....	82
5.3	Discussion.....	87
5.4	Conclusion.....	88
CHAPTER SIX: Conclusion		89
6.1	Research Summary	90
6.1.1	Summary by Objectives	92
6.2	Future Work.....	93
6.3	Conclusion.....	93
Bibliography.....		95
Appendices		102
	Appendix A.....	102

List of Figures

Figure 2.1: The Gettext Based Localization Process (Frimannsson, 2005)	23
Figure 2.2: An example of a poEdit window during a translation session.....	24
Figure 3.1: Software Prototyping Life Cycle (Carr & Verner, 1997).....	29
Figure 3.2: Dwesa Network Infrastructure (Tarwireyi, Terzoli, & Thinyane, 2007).....	34
Figure 3.3: Shop Owner Functionality Diagram.....	36
Figure 3.4: Authorized User Tasks and Role	37
Figure 3.5: LAMP System Architecture	38
Figure 3.6: Viewing Uploaded Shop Items (Njeje, 2007)	42
Figure 3.7: Viewing Uploaded Shop Items.....	43
Figure 3.8: Image Slideshow	44
Figure 3.9: Example of the Edit in Place Functionality	45
Figure 3.10: Uploading New Shop Items	46
Figure 3.11: Viewing Customer Pending Orders.....	47
Figure 3.12: Navigation Support Tool for Shop Owner Assistance.....	49
Figure 3.13: GNU Gettext Localization Process (Frimannsson, 2005)	50
Figure 3.14: The Structure of a PO file.....	51
Figure 3.15: Color Picker	53
Figure 4.1: Sortable Div Elements.....	72
Figure 5.1: Statistics of Background Color Preferences by Questionnaire Participants	84
Figure 5.2: Statistics of Font Color Preferences by Questionnaire Participants.....	84
Figure 6.1: Pencil Cases sold to the SAICSIT 2008 Conference	91

List of Acronyms

AJAX	Asynchronous JavaScript and XML
CAT	Computer Aided/Assisted Translation
CSS	Cascading Style Sheet
FOSS	Free and Open Source Software
G11N	Globalization
GNU	GNU's Not UNIX
GPS	Geographic Point System
GUI	Graphical User Interfaces
HCI	Human-Computer Interaction
I18N	Internationalization
ICT	Information and Communication Technology
ICT4D	Information and Communication Technology for Development
L10N	Localization
LAMP	Linux, Apache, MySQL, and PHP, Perl or Python
LISA	Localization Industry Standard Association
MO	Machine Object
MT	Machine Translation
PHP	Hypertext Preprocessor
PO	Portable Object
POT	Portable Object Template
SMEs	Small and Medium Enterprises
SMS	Short Message Service
TM	Translation Memory
VSAT	Very Small Aperture Terminal
WIMAX	Worldwide Interoperability for Microwave Access
XLIFF	XML Localization Interchange File Format

CHAPTER ONE

Introduction

1.1 Introduction

The adoption of Information and Communication Technology (ICT) applications and services for rural development in Africa plays a significant role in the economic growth and poverty reduction in rural disadvantaged areas (Organisation for Economic Co-Operation and Development - OECD, 2005). However, limiting these people to using ICT applications in a foreign language makes the ICT adoption expensive, difficult and a long process (Osborn D., 2005). Adaptability, sustainability, and usability are some influential factors that affect the growth and impact of ICTs deployed in rural marginalized areas due to cultural differences and needs. Lack of computer skills and access to computers, cultural and linguistic barriers and cultural diversity are also some of the challenges that affect the support of deploying and maintaining ICT related software for rural communities (Ngcobo & Herselman, 2007). Furthermore, the ICT adoption process often takes a long period of time thus those in marginalized areas will only benefit from ICTs over time (Organisation for Economic Co-Operation and Development - OECD, 2003).

The introduction and development of localized content for ICT applications and software typically improves the adaptability of its services for easy sustainability when put into service in a different environment with a different audience. This also improves the usability of the audience who will make use of the localized ICT applications and services. By improving the system usability, the applications and services will be easier to learn and remember. In a multicultural and multilingual environment, failure to adapt to the modes of expression (i.e. language, signs, symbols, etc.) and culture of different stakeholders may reduce the level of benefit of e-Commerce initiatives (Dyakalashe, Muyingi, Terzoli, & Thinyane, 2008). This chapter introduces the motivation for pursuing this research and explains the reason why the research focuses mostly on rural marginalized environments rather than urban environments.

1.2 Research Problem Statement

Software development for rural disadvantaged and marginalized environments may be different from software development for an urban community because of the advanced use of

foreign dominant languages (Osborn D. Z., 2004). This is because when developing software for rural development one has to consider influential factors that may affect the success of ICT project deployment in rural environments. Some of these factors include adaptability, scalability, sustainability and usability. Another key issue is the fact that there are a great number of illiterate and semi-literate people in rural environments as compared to those in urban environments (Harris, 2004). These factors form the core investigation of this research. In this research, illiteracy refers to the following three groups of people:

- ❖ People who cannot read, speak, and write in English but can read, speak, and write in isiXhosa proficiently (referred to as non-English speakers).
- ❖ People who can speak but cannot read or write in English (referred to as Second Language speakers).
- ❖ People who cannot read or write in any language (referred to as Totally Illiterate).

Hence, the term “illiteracy” may have a different context throughout this research; the author opted to implicitly define the word “illiteracy” based on the groups explained above.

The development of e-Commerce platforms and Small and Medium Enterprises – SMEs in rural disadvantaged communities opens up opportunities for employment, poverty alleviation, economic as well as social growth, etc (Golding, Donaldson, Tennant, & Black, 2008). However, ICT adoption in African languages for rural development is still a significant exploration area that can, in future, provide a solution for ICT research projects and applications that are still undertaken from a predominantly Western world perspective (Dyakalashe et al., 2008).

One research endeavor, referred to as the Siyakhula Living Lab project (“Siyakhula” means “*We are growing together*” in isiXhosa), has been deployed at Dwesa, a rural community located on the Wild Coast of the former homeland of Transkei in South Africa’s Eastern Cape Province (Siyakhula Living Lab, 2008). The aim of the Siyakhula Living Lab project is to “*develop and field-test the prototype of a multi-functional, distributed community communication platform for deployment in marginalized and semi-marginalized communities in South Africa*” (THRIP Applications, UFH CoE, 2008; Siyakhula Living Lab, 2008). An e-Commerce platform or shopping e-mall system has been deployed at Dwesa. The artifacts being sold on this e-Commerce

platform are beaded crafts ranging from necklaces to anklets. The shopping e-mall system was developed to use Paypal as the payment method with reason being its low cost, high security measures, ease of use and flexibility (Njeje, 2007) (Paypal, 2009). The crafters, or shop-owners, designing and selling these artifacts are people who are educationally challenged and only have some basic computer literacy background. Most of the group members are elderly women who are in an age bracket ranging from 50 years and above (Dyakalashe, Muyingi, Terzoli, & Thinyane, 2008).

Before the localization and re-design of the shop-owner interfaces, which is the result of the work done in this research, the shop-owners were using interfaces that were semi-translated into English and isiXhosa, with English being the dominant language of communication and interaction. The process of translation was not well disciplined as the translations were embedded in the scripts. Because of this, the shop-owners were given training sessions on how the interfaces and its functionality operate.

The system administrator registers a shop for a certain group of crafters and the crafter can thereafter monitor their registered shop on the shopping e-mall system by themselves. Due to the lack of an educational and computer background, it was a challenge for shop-owners to monitor their online stores because of cultural and linguistic barriers encountered when using the interfaces. Thus, this requires adapting and suiting the interfaces to the shop-owners' cultural and linguistic needs.

1.3 Research Objectives

The e-Commerce platform deployed at Dwesa aims at motivating the local entrepreneurs to engage in trade activities and only supports three types of users: the administrators, shop-owners and customers (Dalvit et al., 2007). The objectives of this research are as follows:

- ❖ **ICT and L10N:** developing culturally and linguistically localized shop-owner interfaces for the local entrepreneurs in rural marginalized environments such as Dwesa thus improving their usability.

- ❖ **Cultural Diversity:** designing adaptable, sustainable, and usable shop-owner interfaces so that the software discussed in this research can be extended to other provinces and regions of Africa with similar backgrounds as Dwesa and different cultural backgrounds. This will improve the growth of ICT research projects and the adoption rate in rural marginalized communities.
- ❖ **User Specification:** the author of this research is a Xhosa, both culturally and linguistically, who has since birth grown up and still lives in the rural Xhosa community in the Qurhu district between East London and King William's Town, Eastern Cape, South Africa. The authors' experience and understanding of the Xhosa beliefs and customs has helped with personal decision-making and self-analyses for re-designing and re-implementing the shop-owner interfaces and good rapport with the shop-owners. Since Dwesa is far from the institution where the author carried out this research, it was difficult to have regular trips to the research site to collect valuable research information. Sometimes, even when a training session or meeting is arranged with the shop-owners, issues such as lack of transportation and bad roads, force the meetings to be rescheduled for another day or trip. The author's decisions include the interface design elements such as colours, content layout, icons, images, language, text or symbols, shapes, etc. Because of the design analysis and decision made personally by the author, there is a need to conduct a light user requirement's elicitation. The elicitation is for observing and studying the shop-owners' requirements for their interfaces based on their cultural and linguistic needs. The data collected from the elicitation will determine whether any changes should be implemented on the shop-owner interfaces. The elicitation will be conducted in the form of user interviews and questionnaires. Furthermore, the aim of the interviews and questionnaires is to determine the effectiveness of and need for cultural and linguistic localization.
- ❖ **Self-reliance:** re-implementing the shop-owner interfaces to improve usability so that the shop-owners can easily manage their registered online stores by themselves.
- ❖ **Interaction and Communication:** implementing a unidirectional audio (i.e. in the research using audio clues) communication with the shop-owners through speaker phones. It is

commonly experienced that audio or video communication can expand the knowledge and understanding of using interfaces better than translated scripts (Majewski & Kacalak, 2005).

- ❖ **Sense of Ownership:** There are currently two registered online shops on the shopping e-mail system; the Ngwane and the Ilingelihle Art and Craft group which are groups composed and formed of elderly women. Both of these groups earn their living by making and selling their crafts. The groups lack the necessary awareness and skills to market their artifact to the global market. The aim of introducing the e-Commerce system was to help them market their artifacts and reach the global market. It is, however, difficult to educate elderly people on the benefits of using e-Commerce systems as they are often uncomfortable with using such a system for marketing their artifacts as they believe in their traditional marketing strategy which is selling the artifact themselves. The community is more comfortable with conducting cash transactions rather than using systems like Paypal and online payments which they do not understand and view these as a way in which someone can rob them of their product. There is a need to explain to these groups the advantages of using this system and to train them on the functionalities of the system so that they can manage it on their own. This makes it important to design shop owner interfaces that they can easily use and understand.

1.4 Research Context

This research focuses on the cultural and linguistic localization of shop-owner interfaces towards developing an adaptable, sustainable, and usable e-Commerce platform for deployment in rural disadvantaged areas. The Localization Industry Standards Association (LISA) defines *Localization* as the process of adjusting and customizing the features of a software product to reflect the cultural and linguistic needs of the targeted area (country or region) thus improving the usability of the software (Esselink, 1998). However, we have to bear in mind that localization is different from translation. Translation is just one of the activities or processes undertaken in the process of software localization. In this research the targeted area is the Eastern Cape province and the software is field-tested at Dwesa.

There is a term “*Personalization*” that is usually confused with localization. Personalization describes a method that limits a software design to a user’s individual requirements (Pluke, Peterson, Pollard, & Szalai, 2006). According to Pluke et al. (2006), “*personalization in relation to language or cultural preferences is usually restricted to letting the user select from a range of options for language and other culturally dependent factors*”. The main difference between the two techniques is that product localization is required when developing a product for a group of people or cultures whilst personalization is often required for product individuality. An example of a personalized interface is a personal website such as the formerly known Google Personalized Homepage, now iGoogle (iGoogle, 2009) (MacManus, 2007). The cultural design requirements used initially in redesigning the shop-owner interfaces were based on personalization because the author designed the interface from personal understanding of the Xhosa culture. However, after successfully conducting the elicitation and finding that changes need to be made, the interfaces will no longer be personalized but localized.

When further expanding and reproducing the software developed in this research to other regions (e.g. other South African provinces), we have to consider the cultural diversity and differences in languages. This introduces the term *Internationalization* (often abbreviated as *I18N*) which according to LISA is “*the process of generalizing a product so that it can handle multiple languages and cultural conventions without the need for re-designing. Internationalization takes place at the level of program design and document development.*” (Esselink, 1998). In the case of this research project, “*internationalization*” refers to the development of multicultural and multilingual user interfaces to support different users with different cultural backgrounds.

After a software product has been successfully internationalized and localized, the software designers can also target the global market. If the software has been internationalized and localized, the process of *Globalization* (G11N) is a lot less difficult to implement. In this research we have combined all these processes to accomplish the implementation of easily adapted, maintained, and understood shop-owner interfaces when replicated in another region.

1.5 Thesis Outline

The ensuing chapters of this thesis are structured as follows:

Chapter Two: this chapter reviews the literature behind the research title and other work related to this research. It discusses the introduction of audio, cultural and linguistic localization in the context of ICT related research project adoption in rural marginalized environments for rural development. The chapter also introduces some of the issues that affect the ICT adoption success in rural marginalization communities such as adaptability, sustainability, and usability.

Chapter Three: in this chapter, we discuss the methodology used in the research and explain the steps of the research model. The chapter presents the introduction to the background of the shopping e-mall system developed by Njeje (Njeje, 2007). The system architecture is also described in this chapter also in addition to a description of who uses the system, their user roles, and how the system operates. The chapter presents the shop-owner interface designs and discusses their functionality.

Chapter Four: this chapter discusses the implementation of the shop-owner interface design and functionality. It discusses the implementation of the research interventions such as colour adaptation, language selection, navigation support, etc. using the Linux, Apache, MySQL, PHP (LAMP) system architecture discussed in Chapter Three along with other programming languages.

Chapter Five: reviews the analysis, findings and significance of the interviews and questionnaires conducted at Dwesa. This chapter discusses the analysis and findings based on the developer's and shop-owner's perspective of the development and usability of the shop-owner interfaces and the implemented interventions.

Chapter Six: this is the final chapter of the thesis and concludes the discussion of the thesis. This chapter continues by discussing the possibility and suggestion of future research and extensions that can be carried out to improve this research.

CHAPTER TWO

Literature Review and Related Work

2.1 Introduction

The use of Information and Communication Technologies (ICTs) in South Africa is increasing, with many projects supporting the implementation of ICTs in rural marginalized environments for rural development. The aim of introducing these technologies to the marginalized communities is to minimize or eliminate the existence of the digital divide and the risk of marginalization which is caused by many factors, including limited literacy. Most members of the population in rural communities, as compared to urban communities, are educationally challenged and lack the resources to further improving their knowledge, particularly with regard to issues of language or linguistic barriers (Osborn D., 2007). According to Osborn (2007), a high percentage of this is composed of women as opposed to men. Furthermore, poverty reduction is, most often, related to gender-equality hence a majority of poor people have been found to be women (Giannini, 2004).

There are eleven official languages in South Africa and those include Afrikaans, English, Ndebele, Northern Sotho, Southern Sotho, Swati, Tsonga, Tswana, Venda, Xhosa and Zulu. English and Afrikaans are the most widely used languages for communication in private and public schools and tertiary institutions in South Africa. Since South Africa is a multicultural and multilingual nation, with a higher rate of non-English speakers in rural areas compared to urban areas, adapting the ICT services and access to a language proficient to the rural communities would be the first step in improving the effectiveness and usability of the technologies. In Chapter One, it is mentioned that the aim of this thesis is to re-design and re-implement the e-Commerce platform that was deployed for marginalized communities in the Eastern Cape (Njeje, 2007) by adapting to the culture and language of the targeted community using the Internet as the source of communication.

Lack in adaptability and sustainability of information products such as e-Commerce platforms are the most common influences encountered when deploying ICTs in rural environments (Rajesh, 2003). In some rural communities, in which ICTs have been deployed, community members are limited in their use of ICTs because of the use of foreign languages. This makes the ICT adoption process long, difficult and expensive. Adapting a product or software

application would determine the success of deploying sustainable ICT projects for rural development so as to make ICTs more accessible and relevant to their users. The process of adapting a product or service for a targeted community or audience with respect to culture and language for better usage and efficient is referred to as localization (Sun, 2001).

In this chapter, the review of literature in the localization process is divided into two sub-levels. The first sub-level is referred to as the surface level (linguistic) localization and it discusses the needs for adapting the features of a market product or items considering the language elements that describe it, like translation, punctuation, dates, weights, measurements, addresses, currency, and so forth, to the needs of the targeted community. The second sub-level is based on cultural localization and covers the adoption of a product developed for one culture to the needs and outlooks of another (Keniston, 1997). The following section of this chapter discusses the concept of localization in relation to ICT for rural development and the need for localizing the ICTs being deployed in rural marginalized environments.

2.2 ICT and Localization for Rural Development

The use of ICTs is growing all over the world as they improve access of information from educational and learning facilities and the economic growth particularly in marginalized environments. There are some problems which are associated with the deficient growth of ICTs for rural development and failure to focus on solving these problems can be costly thus affecting the growth and effectiveness of ICTs in marginalized communities. Some of these problems, among many others, are the cultural and linguistic, socio-economic, socio-political and the technical factors (Rajesh, 2003). The focus of this chapter is on the cultural and linguistic problems encountered when deploying ICTs in rural marginalized communities which pose a threat to the adaptability, sustainability and usability of rural ICT projects.

There have been various efforts at deploying ICT-related services such as e-Commerce, e-Government, e-Health, e-Learning, and e-Judiciary, but most of them have not been targeting ICT development in rural areas (Jakachira, Muyingi, & Wertlen, 2008; Njeje, 2007; Scott, Muyingi, & Thinyane, 2008). However, despite these efforts, there are quite a few projects

which have been implemented in remote rural marginalized communities because of the inadaptable and unsustainable applications of the ICT-related services mentioned above and this is a disadvantage for developing countries that target developing and empowering their rural communities for social and economic development. According to Osborn (2004), not much work has been done to introduce the use of African languages in ICT in addition to which software localization to the African context has not been implemented as ICT services and application adoption and adaption is still oriented towards the Western culture.

E-commerce platforms are one of the ICT-based services that can alleviate poverty in rural communities by assisting rural entrepreneurs (e.g. artifact crafters and small farmers) to interconnect with other electronic markets using the Internet as a source of communication (Mehta & Kalra, 2006; Njeje, 2007). As mentioned earlier, cultural and linguistic issues influence the effectiveness and usability of an ICT-related platform such as e-Commerce and thus affecting and delaying the development and sustainability of ICTs in rural areas. The term culture is complex and has many definitions from different researchers but according to Hofstede's (1997) definition, "*Culture is the collective programming of the mind which distinguishes the members of one group or category of people from another*". Hence, culture influences the behavior of consumers and therefore the ways in which information can be communicated to the audience while at the same time one cultural group would differ from another with regard to the ways in which information is communicated (Ferreira, 2002).

It is commonly experienced that humans are natural experts in social interactions and will communicate easily and effectively with other community members using their mother tongue (Duggan & Deegan, 2003). For people to communicate, a common and understood language must be used to accommodate everyone engaged in the conversation. There are different methods of communication which can be used, the communication can be in an auditory or verbal form (e.g. speaking) or non-verbal form (e.g. writing) (Pretorius & Bosch, 2003). In the case of ICT projects being deployed in rural environments, some ICT-related services still operate, interact and communicate in Western languages, like English, and it remains the dominant language (Ó Slatara, 2002) (Osborn D. Z., 2004). This is a problem for communities

residing in rural environments who would like to benefit from the use of ICT applications and services. It is a fact that some rural community members are non-English speakers and/or may not have been given the chance to go to school to get an education. Due to limited computer education and literacy, the African continent is regarded as the continent most affected by the digital divide, therefore raising the necessity for facilitating the use of African languages in ICTs (Osborn D. Z., 2004). This leads to localizing software and web content developed for rural communities to African languages to aid in allowing people to use ICTs which are fully functional in their preferable language.

2.3 Dwesa Shopping Mall

An e-Commerce prototype by means of a shopping e-mall has been deployed at Dwesa, a rural community located on the Wild Coast of the former homeland of Transkei in the Eastern Cape Province, of South Africa. The Dwesa region has a nature reserve which acts as a tourism catalyst and the community members are talented in the art of designing traditional and cultural artifacts. The traditional artifacts they make are beaded crafts ranging from necklaces to anklets (Dalvit et al., 2007; Njeje, 2007). The crafters who design these artifacts are educationally challenged and lack computer skills and are older people ranging from 50 years in age and above (Dyakalashe et al., 2008). There are some problems that the Dwesa Arts crafters experience such as the skill and infrastructure to advertise and sell their artifacts outside of their Dwesa region.

The shopping e-mall provides an infrastructure that can be used by these arts and crafts designers to advertise their artifacts online for buyers all over the world to view and start buying their beaded art work. The shopping e-mall, along with the Dwesa nature reserve, could also help promote tourism in the region. The shopping e-mall platform supports three types of users which include the system administrators, the shop owners or crafters, and the customers (Dalvit et al., 2007; Njeje, 2007). All these users have different system permissions allowing a user to perform certain functions. The shop owners or rural community crafters are responsible for uploading new items into their shops and making sure that the shop details are updated. This means that shop owners are supposed to self-monitor their registered virtual online stores.

The customers are able to use the front-end (or customer's end) of the shopping e-mall for browsing in English and the administrator and shop owners are able to use the back-end of the shopping e-mall which can be accessed through logging in. The front-end can be used by all users without requiring the user to log in. The system administrator is the only user with full permissions to use the back-end and to register new shops on the platform.

The first version of the shop-owner interfaces for the shopping e-mall system was designed and developed by Njeje (2007) but was not quite usable by the art-craft makers due to their inexperience in using an advanced computer system and their inability to read and write in different languages rather than their mother tongue. The first version of the shop-owner interfaces for the shopping e-mall system was semi-translated to isiXhosa, but fully operational in English, as the media of communication and interaction to perform certain shop-owner functionalities. Because of the low level of educational background of the crafters, this required some training sessions and demonstrations to be conducted for the crafters to understand how the shop-owner interfaces operate. The training sessions and demonstrations were conducted at the local schools where an ICT infrastructure (computer living labs) has been deployed. Another problem, apart from the fact that the crafters do not know how to speak or write in English, is that some of them are also unable to read or write in their own language which is isiXhosa. For this reason, the training sessions had to be conducted in isiXhosa which is the language they speak. This was one of the constraints as the shop-owners were supposed to monitor, maintain and sustain their stores from a system that was foremost represented in English, a language they can neither speak, read, nor write.

Due to the above mentioned limitations (i.e. translation, usability, and user-friendliness), the initial version of the shop-owner interfaces was not compatible with the low educational conditions and lack of computer literacy encountered at Dwesa. This led to the need to develop a new localized version of the shop-owner interfaces for the shopping e-mall system which enables the ease of use and user friendliness of the interfaces for the art-craft makers. This research is the result of work done in the localization of the second version (i.e. localized version) of the shop-owner interfaces for the shopping e-mall system leading to the

introduction of the topic “*Localization*” to be discussed later in this chapter. The following section discusses the term “*Usability*” and the investigations performed in relation to the improvement of the use of computers and the Internet, relating to the context of this research.

2.4 Culture and Usability

Earlier in this chapter, Hofstede’s definition of culture is used. However, in Human- Computer Interaction (HCI), the definition of culture is often referred to as “*the common values, attitudes and behavioral patterns shared by a group of people*” (Vatrapu & Pérez-Quñones, 2006). According to Vatrapu & Pérez-Quñones (2006), these common values, attitudes, and behaviours can influence the selection of colour, language, shape, symbols, and so on during the development of a specific product or software hence development processes like localization, globalization, and internationalization have been proposed.

Similar to the definition for culture, the term usability is complex and has no common specific definition. According to the International Organization for Standardization (ISO 9241, 1998), usability is defined as “*the extent to which a product can be used by specific users to achieve specified goals with effectiveness, efficiency, and satisfactory in a specified context of use*”. Ford (2005) defined usability in six terms, or measurements, which read as follows:

- ❖ **Learnability:** is the ease with which targetted product users learn how the product services are used. According to Ford (2005), there are different ways of indicating and assesing learnability, however, the most accurate metrics used are by measuring: (i) the number of users making the same error, (ii) the number of users completing a task successfully, (iii) the number of errors made by novice users, or expert users, or the ratio of these errors, (iv) accuracy and speed.
- ❖ **Memorability:** refers to the ease of remembering how to use the system after it has been learnt. According to Ford (2005) and Nielsen (1993) memorability is best measured by assesing the performance of casual users who have not made use of the system for some time and their accuracy and speed during the re-use of the system.

- ❖ **Efficiency:** is the degree to which the system supports users with their tasks. Furthermore, only once they have learned the system is productivity possible (Nielsen, 1993). According to Ford (2005), efficiency of use is measured in the accuracy and speed of the user to complete a task.
- ❖ **Safety:** refers to the reduction or prevention of system errors that affect the performance measurement of accuracy (Ford, 2005). Safety is measured in terms of successfully completing a task and performance speed.
- ❖ **Satisfactory:** refers to the users' comfort and probability towards the use of the system (Forkjaer, Hertzum, & Hornbaek, 2000). According to Forkjaer et al. (2000), satisfaction can be measured with SUMI (Software Usability Measurement Inventory) which is a 50 item questionnaire for measuring likability, efficiency, helpfulness, control and learnability (Kirakowski & Corbett, 1993).
- ❖ **Utility:** is the ability to provide the proper system functionalities which reflect what the users want and need.

Culture has an impact on peoples values, attitudes, and behaviours thus it is imperative that software developed for a global market should reflect the cultural orientation of its intended users as software translation is not satisfactory (Khaslavsky, 1998). According to Khaslavsky (1998), *“conceptual localization that snugly fits a user’s cultural specific mental model of the software with its functionality, feedback, and support for learning is a much more effective way to design software for international use”*. Therefore, as we mentioned above the six measurements that defined, whenever a system provides support for learning it helps the improve the understanding of the targeted users on how to use the software.

The methods and techniques for collecting these measurements involves both qualitative (i.e. analysis and observations) and quantitative (i.e. questionnaires and interviews) aspect. The data collection will be discussed in the following chapter. The following section discusses the introduction of culture and usability into the proposed software development proccess referred to as Localization.

2.5 Localization

Software development for rural communities may differ in terms of usage and consumption from that of urban communities due to many influential factors including low literacy rates and cultural mismatches resulting in rural marginalization (Ngcobo & Herselman, 2007). These two factors have great impact on the success of rural development of rural areas using ICTs. This is because most rural African communities are educationally disadvantaged and lack computing skills. Another reason is the fact that they are not exposed to what is referred to as the “*real world*”, and lack of ICT infrastructure is another factor. According to Osborn (2004), localization of software and web-interfaces to accommodate illiterate and semiliterate users in rural areas by using African languages can eliminate the risk of rural marginalization hence developers should consider the localization of ICTs and the services they provide.

Localization (with the standard abbreviation L10N - meaning that there are ten letters between the letters “l” and “n”) is the process of adapting, translating and customizing a product or service (e.g. software applications and web documents) written and designed in the linguistic needs and cultural outlooks of one culture to the linguistic needs and cultural outlooks of another culture (Dohler, 1997; Keniston, 1997). The LISA also has a similar definition of localization which is making a product linguistically and culturally suitable for a specific targeted environment, be it a country or rural area. As mentioned before, there are two main stages of the localization process; the cultural localization process and the linguistic (technical) localization process. The following sections explain the two localization processes respectively and finally explain the use and importance of audio in localization.

2.5.1 Cultural Localization

Differences in culture influence the development and usability of an ICT software product developed for a multicultural and multilingual audience. This is because the requirements of a software product to be developed for a specific cultural group may differ for another cultural group. Hence, according to Keniston (1997), properly culturally localized software can be easily adaptable to different cultural groups. For cultural localization of software or web-interfaces, a

designer or developer should identify and understand the cultural differences and specific requirements within a culture (Masoeu & de Villiers, 2001). This means culturally localizing software involves adjusting and adapting the software to reflect the local customs, concepts and habits of the targeted area for deployment. These cultural differences in web-interface design are commonly referred to as cultural markers and are used to improve the ease of use of an interface. Cultural markers refer to the interface design elements and cultural elements or common knowledge shared within a culture of the targeted audience (Sun, 2001). There are various categories of cultural markers and they may be visual or auditory, but some include colours, content layout, icons, images, text or symbols, shapes, metaphors, etc (Barber & Badre, 1998). Barber & Badre (1998) define the merging of culture and usability as “*Culturability*”. The use of cultural markers in web design is important especially if it is designed for a multilingual and multicultural rural community. Failure, on the part of the system designer, to understand the cultural differences may lead the user to make unnecessary mistakes due to misunderstanding.

2.5.2 Linguistic Localization

This stage of the localization process involves the translation of language resources to the language preferably used by the members of the targeted community. These language resources include the adjustment of the system features such as currency, text or symbols, date and time, menus, weight, etc (Weir & Lepouras, 2001). The high rate of non-English speakers in rural areas and their lack of computer skills is the main reason why software and web interfaces should be linguistically localized to the local language used by community members. However, there are also some linguistic problems encountered due to illiteracy that need to be addressed. Some rural community members can speak but cannot read or write in English whilst others are unable to read nor write even in their own native language and are not proficient in English. Illiteracy makes it difficult for rural community members to interact and communicate with even the simplest of user interfaces (Goetze & Strothotte, 2001). The shopping e-mall needed to be linguistically localized as the targeted audience (Art and Craft makers) has a limited educational background and some lack proficiency in English thus

qualifying them as non-English speakers. Localizing the system to their own language, in this case isiXhosa, will help them understand the functionalities and meaning of the icons, text, images, etc used in the system.

Language is deeply intertwined with culture but in this chapter it is distinguished from culture for simplicity and better understanding. However, the level of cultural orientation in linguistic localization may differ from pure textual translation e.g. standard language, to cultural-related concepts translation. Linguistic translation is a large area of localization and is the most basic section in the localization process, thus many people tend to confuse the relation between the two processes. Translation only includes the conversion of written text to a different targeted language. There are several different types of translation tools that can be used in translation and localization which include Machine Translation (MT) and Computer-Assisted Translation (CAT) (Briggs, 2007). These translation tools will be discussed in brief details later in this chapter. In some cases, translation also includes the conversion of audio or sound to the targeted language which introduces some other kind of localization referred to as Audio or Auditory localization and this is discussed in the following sub-section.

2.5.3 Audio in Localization

Auditory or audio communication is another way of localization different from cultural and language localization. There have been few speech-driven interfaces developed to successfully target the improvement in accessibility for illiterate and disabled users in rural areas due to some challenges relating to the research such as multilingualism and cultural barriers (Turker et al., 2003; Duggan & Deegan, 2003). The path of audio communication can be unidirectional or bidirectional between interfaces and the users. Unidirectional communication is incomplete because it can only be an input or output type of communication. It can either produce audio information output to the user or allow the user input by speech recognition technologies and convert it to text. Although bidirectional audio communication is complex and difficult to implement in multilingual interfaces, it can better improve the communication between the interfaces and users (Majewski & Kacalak, 2006). Because of the complexity and difficulty of implementing bidirectional audio or sound communication, the following chapters in the

research will discuss the implementation of text-to-speech audio or sound communication in user interfaces.

Text-to-speech and speech recognition (i.e. speech-to-text) tools allow a system to communicate with the users that have limited computer literacy by developing audio interactive interfaces (Plauché, Nallasamy, Pal, Wooters, & Ramachandran, 2006) (Heerden, Barnard, & Davel, 2009). Because people mostly communicate and interact through their social life with the aid of speech, such voice enabled systems will allow users to understand computer systems and ease of accessibility to ICTs (Duggan & Deegan, 2003). Using these speech-driven tools or technologies would improve the usability of a system by improving the communication and understanding between the user and the system interfaces. This means that the audio or sound files would be localized and translated to the language understood by the targeted community. As said earlier in the chapter, some rural community members do not know how to speak and/or write in English and, in addition to this, some may not be able to read or write even in the native language. However, developing audio or sound interactive interfaces overcomes this problem because a user would not have to read but listen to the audio or sound instruction output to perform a set of tasks in the text-to-speech model. Also, with the aid of speech-to-text transformation the users are able to write and input into the system (e.g. speech-to-text can be used by the shop owner when uploading item details into the system).

2.6 Localization and Translation Tools

There are several types of free and open source software (FOSS) based translation tools which can be used by different translators. The translator chooses the correct tool to use based on the requirements of the software to translate and the type of the platform (platform dependent). This means that the translator also has to choose a translation tool that will be compatible with the platform upon which the software will be used i.e. the operating system or programming language. Translation tools are also chosen based on how fast or quick they can translate, for example a web interface or document. These translation tools are grouped into two categories, namely; TM (Translation Memory) and MT (Machine Translation) tools (Beshar, 2008). Translation Memory tools are usually classified under the CAT (Computer-Aided/Assisted

Translation) category. The differences between these categories are discussed in the following subsection and in addition some translation tools will also be introduced and explained according to the category to which they belong.

2.6.1 Translation Memory and Machine Translation

This type of language translation technology enables a human translator to keep recorded words and stores translated words, sentences, paragraphs, or phrases of documents in a database (Translation Memory, 2008). A Translation Memory system stores words and sentences in pairs that have already been translated in the database for future repeated use and easy searches. A pair consists of two segments called translation units, the source language segment (original non-translated text) and the target language segment (translated text to the targeted language). When a document is edited or updated and the new edited text matches with some text in the source language segment, the system will find the translated version of the text and assign it to the new edited text or sentence. Hence, during searches, the translation memory system suggests all text or sentence matches found in the database (Ó Slatara, 2002). All targeted language segments without a match are manually translated and automatically stored in the translation memory. This helps to avoid repeating translation of a sentence that has already been translated by the human translator, thus improving the rate of translation. If a segment cannot find a match in the translation memory, that translation is referred to as being fuzzy and instead looks for a similar segment in the database (Beshar, 2008).

There are two types of translation memory tools, text-based and binary-based tools. Text-based translation memory tools are used to translate text documents e.g. HTML files and may only store the source and target language segment. On the other hand, according to Beshar (2008), binary-based translation memory tools are usually used with binary files such as *.exe* and *.dll* program files. The binary-based tools store more than just the two language segments but may also store embedded errors, location and size. The main difference with MT compared to TM is the fact that it does not require a human translator. MT arranges all the data for the translatable strings in a single program so that translation is possible without a human

translator (Craciunescu, 2004). In the research, the selected translation technique uses a TM and this is because, as explained earlier, not much has been done to localize in African languages. Hence, MT may not be able to translate in African languages and localizing to African languages may require a human translator with the understanding of African languages.

2.6.2 Translation Databases and Portals

The following section provides literature on some of the translation databases and translation portal which have been successfully implemented to achieve well structured language translations.

2.6.2.1 GNU Gettext and PO files

GNU gettext utilities provide a framework to which other free packages, such as PHP, may produce and update multilingual messages using translation databases or files such as textual and editable Portable Objects (PO) files (Drepper et al., 2007). The GNU gettext uses two types of file format, the first being PO file formats and the second being the Machine (or Message) Object (MO) file formats. The PO file format is an example of a database table that stores all translatable strings from the source files in the process of localization. A PO file is made up of many entries in the form of a translation unit where every entry holds a relation between the original non-translated string and the corresponding translation version of that string. A PO file is created for each and every target language. For example, if translating a web document or software into two different languages (i.e. in the case of this research, translating from English to isiXhosa), a PO file will be created for storing strings in English and a separate PO file will also be created for the Xhosa translation version that PO file. However, PO files are a type of translation memory tool which can be read and edited by human translators. Hence PO files are compiled and converted to MO file formats so that the file and its translation can be understood by the machine programs (Drepper et al., 2007; Frimannsson, 2005). The MO file format is a binary form of presenting the PO file or translatable string table used by an application to retrieve the translated strings during runtime. The combination of gettext and PO files is the type of translation technique which is being used in this research.

Figure 2.1 below describes and gives an overview of the process of localization and translation using GNU gettext and PO files and also includes the use of GNU gettext tools used in the process. This chapter does not discuss in detail how the extraction of strings, the translation of strings in the PO file, and the conversion of PO files to MO files is actually performed. That will be discussed in the next chapter dealing with the research methodology and system design of this project.

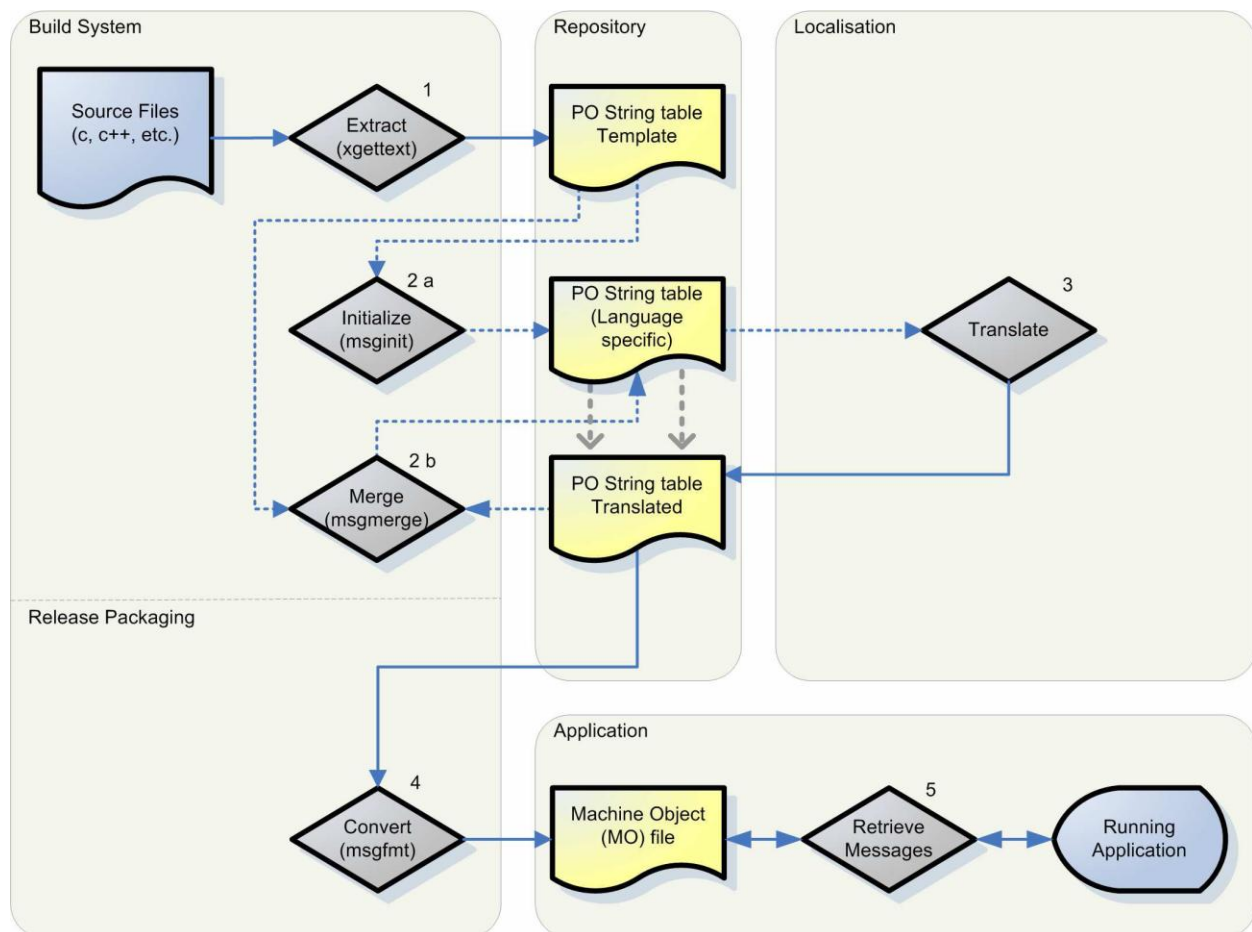


Figure 2.1: The Gettext Based Localization Process (Frimannsson, 2005)

There are different types of translation tools that can be used to edit PO file such as kLabel, poEdit, gTranslator, etc depending on the platform the software to be localized is operating in. All of these translation tools provide the translator with a simple graphical user interface (GUI) that is used to scan the source and extract all translatable strings using a catalog manager. The following figure shows an example of a PO file editor which was used during a translation

session of the research. The forthcoming screenshot (Figure 2.2) was taken during the period of translating the strings and sentences used in the shop-owner interfaces.

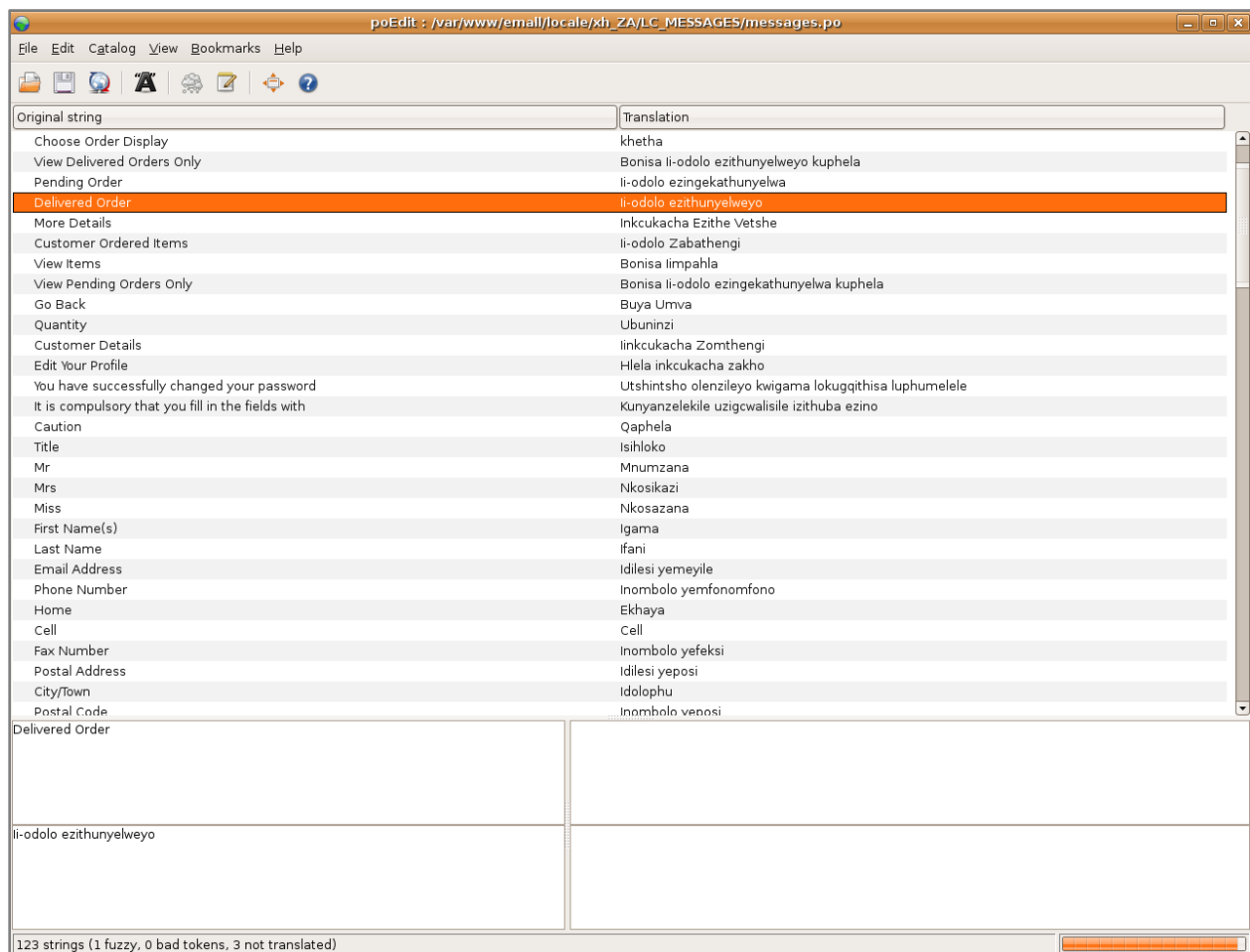


Figure 2.2: An example of a poEdit window during a translation session

This figure displays a sample of a poEdit window during the process of translating the e-Commerce platform. poEdit is the translation tool which was used for editing PO files used in the research and it runs on both the Linux and Windows platforms including other UNIX operating systems (poEdit Homepage, 2005). In Figure 2.2, there is a table with two columns which represent the translation units (the original non-translated string and the translation corresponding to that string) that were explained earlier in this chapter. Some of the features of poEdit include (poEdit Homepage, 2005):

- ❖ Automatic conversion and compilation of PO files to MO files
- ❖ Adding comments on a translation unit

- ❖ Built in spelling checker
- ❖ Scanning specified source file for new translatable strings by updating the catalog manager

2.6.2.2 Pootle

Pootle or PO-based, Online Translation/Localization Engine is an online CAT tool (Pootle, 2008). Pootle is a web based PO file translation tool mostly used to host online translation projects for open source software and application. It is an online open-source localization and translation management tool that allows translators all over the world to connect and work together to translate web-interfaces and software on the web (Dalvit et al., 2008). This translation tool makes use of PO files and XML Localization Interchange File Format (XLIFF) to translate projects such as OpenOffice.org and Mozilla and many others (Sasikumar et al., 2005). Pootle was developed under the WordForge project and now is sustained by translate.org.za intending to assist in community based translations. Translation events such as the translate@thon conducted in South Africa have also been successful with the use of this translation tool (Dalvit et al., 2008). The Pootle translation tool has some features such as (Translate Toolkit & Pootle, 2008):

- ❖ Integrated translation memory – which is an offline tool created by the user or translator to store recorded translated words, sentences, paragraphs, or phrases of documents in a database
- ❖ Online XLIFF translation support
- ❖ Glossary management and matching – provides a glossary for each project
- ❖ Creating goals and various user task assigning with specific permissions

2.6.2.3 Rosetta

Rosetta (Launchpad Translation, 2008) is a CAT tool and, like Pootle, it is also a web based open source translation tool. This translation portal allows all previous translations to be stored in a dictionary which can later be used whenever there is a change of the translatable string in the source file(s) (Vanhouwaert, 2002). The dictionary stores and keeps track of all complete text and sentence translations. Whenever there is a change in the source file, unchanged

translatable strings are automatically translated using the dictionary. The dictionary is used to translate other documents with the same text fragments. This reduces the task of redoing the translation process for new, edited or updated sections of the source file(s).

2.7 Conclusion

In this chapter, we have discussed the importance of deploying culturally and linguistically localized systems due to the issues encountered such as adaptability, sustainability and usability. This could also help bridge the digital divide that is the main cause for rural marginalization. There has not been a lot of research done in the effort of localization in the African context and languages targeting rural development, hence there is not too much literature relating to this current research. Developing culturally meaningful, user-friendly and user-attractive interfaces is also of utmost importance especially when targeting users from disadvantaged rural areas from a multicultural and multilingual community.

The next chapter will cover the actual research methodology and system design used to re-develop the shop-owner interfaces of the e-Commerce platform deployed at Dwesa. The shop-owners' end operates in English and isiXhosa as it is still being field tested but mainly intended to be replicated in other regions and rural marginalized communities such as Dwesa, not only in South Africa but possibly elsewhere in Africa as well.

CHAPTER THREE

Research Methodology and System Design

3.1 Introduction

This chapter describes, firstly, the methodology used in the research and, secondly, the process of designing the shop-owner interfaces of the e-Commerce platform that has been deployed at Dwesa. The primary research paradigm utilized in this study is a traditional and practical approach using a Software Prototyping development method. Within this paradigm, the development and implementation of the new system components is designed based on extreme programming. In Chapter Two, we discussed localization as a possibility for improving the usability of systems developed for rural areas which run the risk of marginalization, thus increasing the adoption rate of the provided services for the targeted users. The system has been culturally and linguistically localized into two languages, English and isiXhosa, and to later support localization into languages spoken in other field areas (The localization was done in order to easily support any other language). This chapter will focus on the design of the newly improved shop-owner interfaces. The implementation details will be given in Chapter Four.

3.2 Methodology

Prototyping or Software Prototyping refers to the rapid software or system development to validate requirements and is the process of utilizing prototypes in software or system design (Sommerville, 2000). The prototyping model allows the developer to initially design a simplified trial version of the proposed product which is then examined by the targeted users of the product. The feedback is sent back to the developer for refining the software (Green & DiCaterino, 1998). The prototyping model involves the following steps (Software Prototyping, 2009) which are defined, in context of this research, as:

- ❖ **Identify basic requirements:** In this research the basic or initial requirements for developing the shop-owner interfaces were based on the author's background knowledge of the Xhosa beliefs and customs as the author is from the Eastern Cape and speaks isiXhosa. This information is obtained through the self-evaluation of the system developer.
- ❖ **Develop initial prototype:** The first version or prototype of the shop-owner interfaces is designed to reflect the developer's system requirements.

- ❖ **Evaluate the prototype:** The initial prototype is evaluated by the intended users, in this case the shop-owners, through usability evaluation techniques such as questionnaires, interviewing, and system testing. The target of this step would be to obtain valuable information on cultural differences and additional changes on the first prototype.
- ❖ **Revise and enhance the prototype:** The feedback from the evaluation step can be used in the design of a second prototype which is localized, based on the shop-owner's specifications.

Figure 3.1, below, provides a simple example of the software prototyping life cycle.

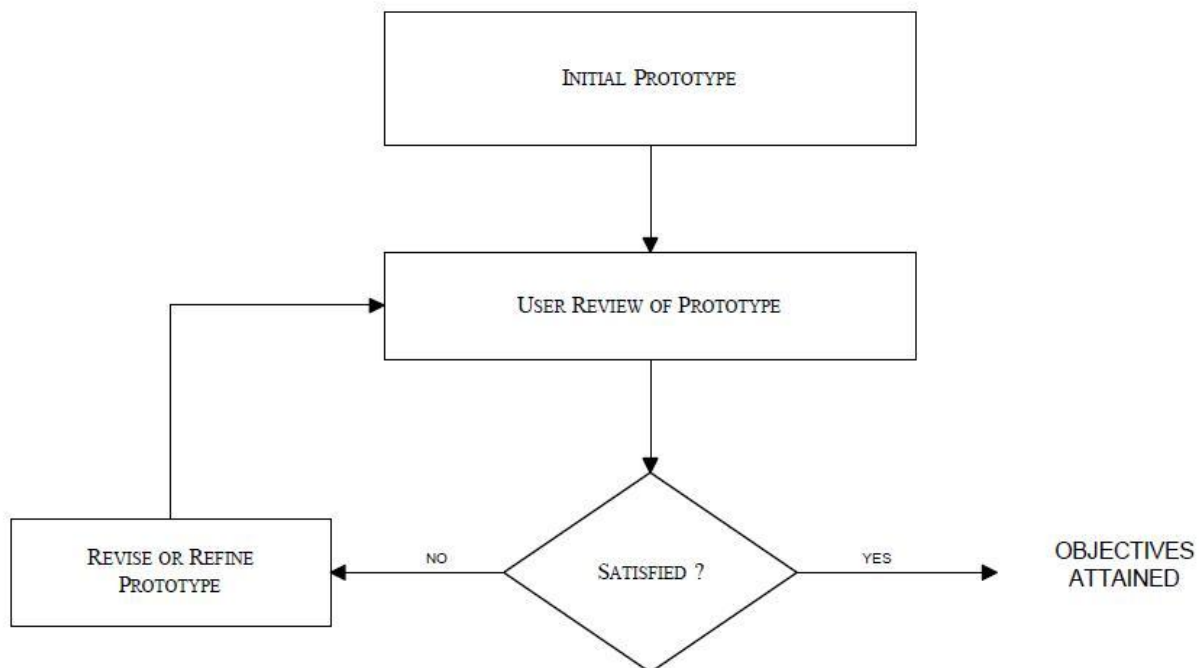


Figure 3.1: Software Prototyping Life Cycle (Carr & Verner, 1997)

According to Sommerville (2000) and Green & DiCaterino (1998), the benefits of software prototyping are as follows:

- ❖ The rapid development of the initial prototype thus allowing the targeted users to evaluate the prototype in the early development stages of its software (Software Prototyping, 2009). This means prototyping provides a fast delivery time.
- ❖ It reduces the time and cost for software modification. This is because without an initial prototype for user requirements evaluation, there would be more changes that the

developer would keep implementing on the product thus taking time and making the development more costly because of these changes.

- ❖ It improves the usability of a system developed under the prototyping steps which thus improves the users' involvement.
- ❖ The functionality of the final product is designed similarly to the specification the targeted users have with regard to the product.
- ❖ It improves the maintainability of the system.
- ❖ It reduces the number of errors encountered by users during their use of the system.

However, using the prototyping model for software development also has its drawbacks (Software Prototyping, 2009) (Green & DiCaterino, 1998):

- ❖ It can lead to user confusion and false expectations of the prototype due insufficient analysis of the other implemented model such as database normalization and testing.
- ❖ Misunderstandings between the software developer and targeted users' design specifications may lead to poorly designed systems.
- ❖ Difficulty of maintaining the targeted users' interest and involvement during the process.

There are two major types of prototyping techniques, namely: Rapid or Throwaway Prototyping and Evolutionary Prototyping techniques. The proper technique is chosen depending on the application domain, user interaction, and whether its development cycle suites that specific proposed development research (Sommerville, 2000). The following section describes the differences and similarities between the two prototyping techniques.

3.2.1 Rapid Software Prototyping

Rapid prototyping refers to the development of a functioning prototype from initial requirements or specifications so that the intended users can then re-examine their requirements from this prototype as an experiment (Software Prototyping, 2009). The prototype is developed based on the identified basic requirements and displayed to the users as an example of what the final system would look like based on their specifications. After the users have completed their evaluation and examination, the prototype is discarded and does

not become part of the final system. For this reason, rapid prototyping is also referred to as Throwaway Prototyping. In addition, the system then continues to be developed, from scratch using a different development process (Sommerville, 2000). The advantage of rapid prototyping is that the initial design of the prototype is a quick and detailed design which saves time and money because of the minor changes that would be done to the end-product (Najjar, 1990). System errors and incorrectly applied specifications can also be detected at an earlier stage (Sommerville, 2000).

Because rapid software prototyping is a quick process, the developer may not get a complete set of user requirements thus omitting or excluding important steps and features. This would lead to a system that has an end-product of poor design quality (Gordon & Bieman, 1993). Another problem occurs when the prototype to be discarded is not actually discarded in the final design because of poor initial planning thus making the system difficult to use and maintain (Software Prototyping, 2009).

The system developed for this research is Free/Open Source Software (FOSS) and supports the possibility of extending and/or replicating its services to other regions with a similar background to Dwesa. For this reason, we cannot conclude the possibility of discarding the initial prototype(s) because of the users change in environment and cultural requirements. Hence, we cannot limit the system to a rapid software prototyping model and also conclude a final end-product.

3.2.2 Evolutionary Software Prototyping

Evolutionary prototyping is similar to rapid prototyping as it also follows the same initiative of initially implementing a working prototype and exposing it to the targeted users for evaluation. However, the difference is that the system is refined through several prototypes of the system until the final system is complete based on the users' design specifications (Ambra & Ruta, 2009). According to Ambra & Ruta (2009), *"There is no better way than trying a requirement before agreeing to it. This is possible if a system prototype is produced by the developer"*.

The advantages of using evolutionary software development, when compared to those of rapid software prototyping, are as follows:

- ❖ According to Anton, Carter, Earp, & Williams (2001), user requirements may lack focus and/or direction due to the clients' constant changing opinions and unclear specifications. However, evolutionary prototyping provides a functional prototype that the users can evaluate from and the more the users use the prototype the more they gain understanding of the requirements (Carr & Verner, 1997). In addition, they can also assist with additional functionalities offered by the prototype.
- ❖ The users' requirements elicitation can be collected during the development process and are not limited to the period prior to the design and implementation of the prototype(s) (Ambra & Ruta, 2009).
- ❖ The initial prototype(s) form all or part the final proposed product (Gordon & Bieman, 1993).
- ❖ Evolutionary prototypes are functional systems which can be provisionally used until the final system is complete (Software Prototyping, 2009). Hence, offering the possibility of extending and replicating this research in the future. This software will be a functional system whilst the users probably wait for an improved prototype with new features proposed by the users (Anton et al., 2001; Software Prototyping, 2009).
- ❖ Evolutionary software development supports the users' system learning ability because of the users' involvement in the modification of the prototype.

This process of evolutionary software development is applied in this research because the initial prototype or version of the shop-owner interfaces is developed upon the users' partially known cultural requirements. The fact that the developer personally understands the customs and beliefs of the Xhosa culture does not entirely imply that the developer understands all the users' requirements, hence a requirements elicitation is required. The reason why the author personally chose the cultural markers instead of collecting this data from the shop-owners themselves is due to the following circumstances:

- ❖ It is difficult to have regular trips to the research site to collect valuable research information pertaining to the cultural markers.
- ❖ Lack of transportation to the meeting venues (usually the schools with installed computer labs) because of bad road conditions (especially during bad weather) which forced the research team to cancel a visit to some areas of Dwesa where the shop-owners reside.
- ❖ Electricity failure when conducting system demonstrations.
- ❖ Computer and/or Network failures.

The evaluation of the initial prototype was, due to these reasons, conducted at a later stage of the system development through questionnaires, system demonstration, and interviews.

3.3 Dwesa ICT4D Infrastructure and Services Overview

The Siyakhula Living Lab project is undertaken by the University Of Fort Hare Centre Of Excellence along with the Rhodes University Centre of Excellence. The project is the first ever to establish a rural WIMAX (Worldwide Interoperability for Microwave Access) network in South Africa that connects to the internet via a VSAT (Very Small Aperture Terminal) back haul. The WIMAX network (Mandioma et al., 2007) is deployed across five local schools in Dwesa and each of these schools has a computer lab. The computer labs in the schools are not only meant for the school teachers and students, but for the entire local community. Figure 3.2 shows the WIMAX local loop network infrastructure deployed at Dwesa showing only four of the five connected local schools.

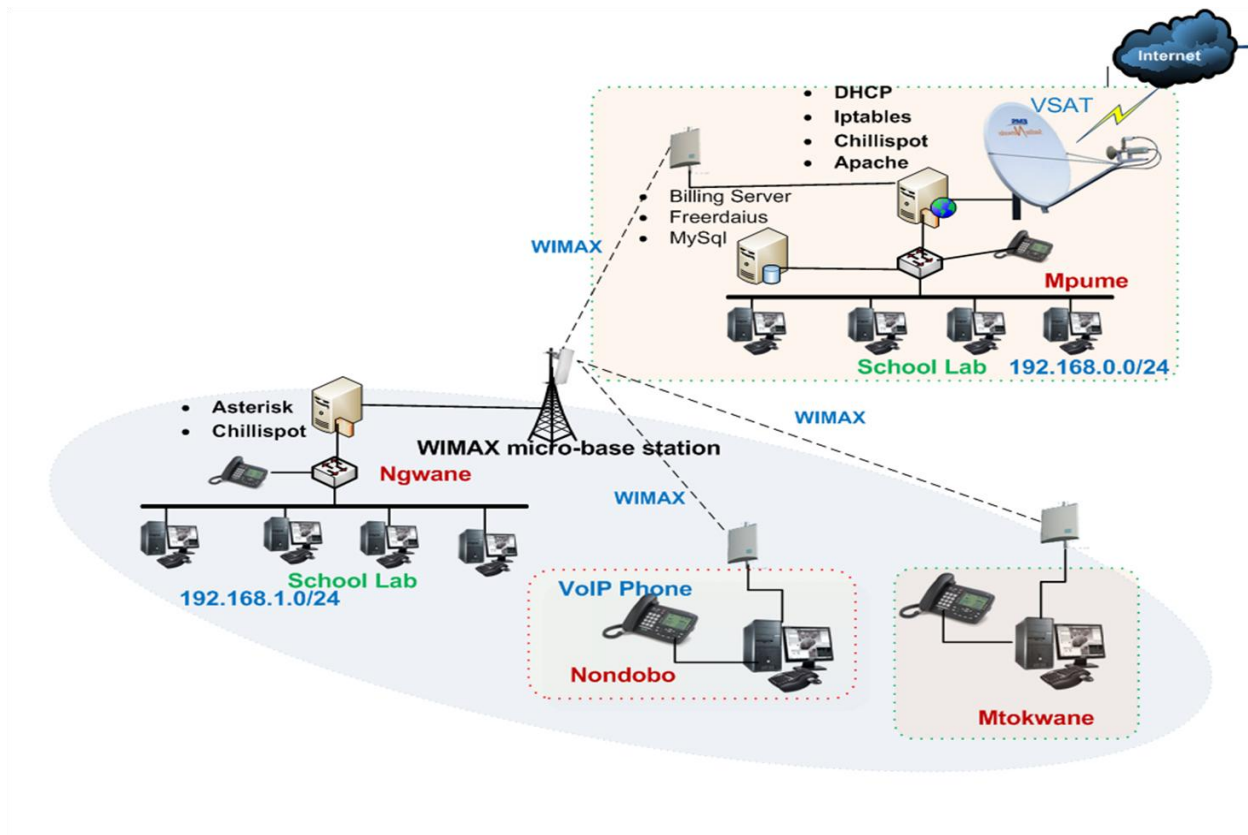


Figure 3.2: Dwesa Network Infrastructure (Tarwireyi, Terzoli, & Thinyane, 2007)

Some of the services that have been gradually developed or which are currently in progress at Dwesa include an e-Commerce, e-Government, e-Judiciary, and m-Commerce platform.

3.3.1 Dwesa Shopping e-Mall Background

The aim of developing the shopping e-mall software was to improve economic uplift and introduce new markets for rural marginalized communities. Some of the findings that the researchers discovered after some field trips and whilst conducting community interviews were as follows (Njeje, 2007):

- ❖ The art and craft groups required an e-Commerce system that they could use to sell their artifacts as they lack the skill and resources to advertise and sell their products in other regions around Dwesa and all over South Africa and other countries in Africa and overseas. The Dwesa shopping e-mall was designed to be self-monitored by the authorized art and crafts groups whose shops will be registered by the system administrator.

- ❖ Some of the community members, including the teachers at the local schools, lack computer literacy skills as they have little experience of using computers.
- ❖ The local art and crafts groups were both computer and educationally challenged and the majority of them are non-English speakers. Most of them are elders above the age of 50 years. Some are dependent on the profit they make out of the products they sell and others on the monthly government welfare grant to sustain themselves. This required that the art and crafts groups attend training sessions on how to use the system and other kinds of training sessions for computer literacy.

Against this background, the Dwesa Shopping Mall (<http://www.dwesa.com>) was developed and deployed. The shopping e-mall deployed at Dwesa supports two types of user interfaces: the customer interfaces and the shop-owner interfaces. The system is a Free/Open Source Software (FOSS) based e-Commerce platform developed using the LAMP architecture due to its adaptability, robustness, and cost effectiveness (Njeje, 2007; LAMP- Linux, Apache, MySQL, and PHP, 2009). The customer interfaces are accessible to any user who visits the shopping e-mall site whereas the shop-owner interfaces are only accessible to the system administrator and the shop owners via a login portal. The shop-owner interfaces, however, had great limitations with respect to cultural and linguistic localization. The translation was an ad-hoc type of translation that would require code modification when a new language was introduced into the system. The interfaces were mostly limited to translation and the shop-owners could not have an option of choosing a preferable language to use. Also the translations in both English and isiXhosa were hard-coded into the scripts. This translation paradigm is not a principled way of translating a software application as the translator might not understand the programming language(s) used to develop the software.

In addition to the technologies used to develop the first version of the shopping e-mall system some of the following tools and scripting languages were used to re-design the system for the shop-owners:

- ❖ JavaScript
- ❖ Asynchronous JavaScript and XML (AJAX)

- ❖ GNU gettext tools along with the use of poEdit as the PO file editor
- ❖ Cascading Style Sheet (CSS)

These tools and scripting languages were used in combination for building certain functionalities that could better suit the needs of the shop-owners in such a way that their interfaces have a cultural meaning, user-attractiveness and, primarily, are both understandable and usable.

Figure 3.3, below, displays the functional diagram for the shop-owners' end.

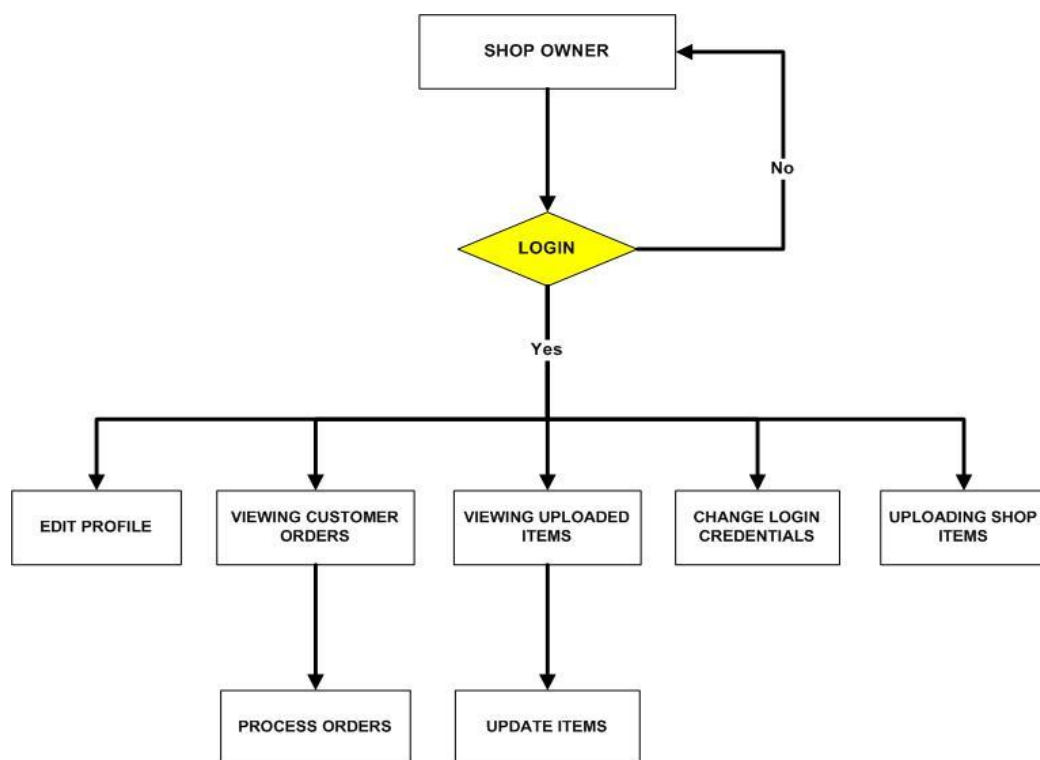


Figure 3.3: Shop Owner Functionality Diagram

Since this research focuses on the improvement of the usability of the shop-owner interfaces the functionality of the interfaces remain the same, thus the shop-owner functionality diagram for the new improved version is the same as the shop-owner functionality diagram for the old version of the shop-owners' end. The difference is how the functionalities are displayed on the shop-owner interfaces.

3.4 System Architecture

The following section discusses the actions and roles assigned to each of the intended users of the shopping e-mall system using the LAMP system architecture.

3.4.1 User Roles

As discussed in Chapter Two, the shopping e-mall system supports three types of users and those are the system administrators, the shop owners, and the customers (Njeje, 2007). The figure below displays the roles that each authorized user can perform on the system. The shop administrator is the only user that has access to all roles of the system. The system administrator is responsible for registering and deleting new users and shops. Naturally, the system administrator could also act as a customer. The shop owner can also play the role of shop administrator and also can be a customer for another shop registered in the shopping e-mall system, and vice versa. Figure 3.4, below, shows the authorized user tasks and roles and which task each shopping e-mall user can perform on the system.

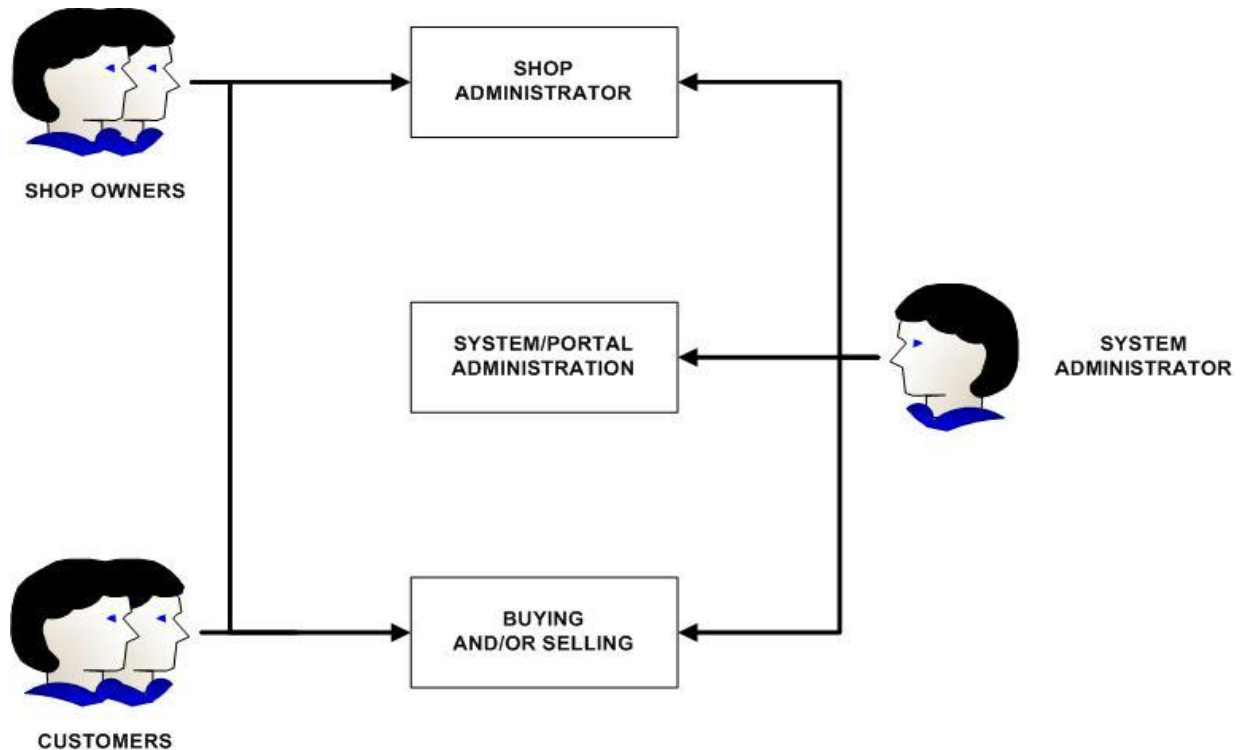


Figure 3.4: Authorized User Tasks and Role

3.4.2 LAMP Architecture

The shopping e-mail system was developed using the LAMP architecture which is displayed in Figure 3.5 below. The shopping e-mail system uses Free/Open Source Software so that it can be replicated at no cost in other rural marginalized communities. The diagram describes the data flow after the user makes an HTTP request. The Apache web server processes the HTTP request and sends an HTTP response back to the user via the web browser. The HTTP request could be a request for accessing the MySQL database using MySQL queries or accessing the mail server using the SMTP protocol. The requests are executed by the PHP programming language using HTTP GET/POST request.

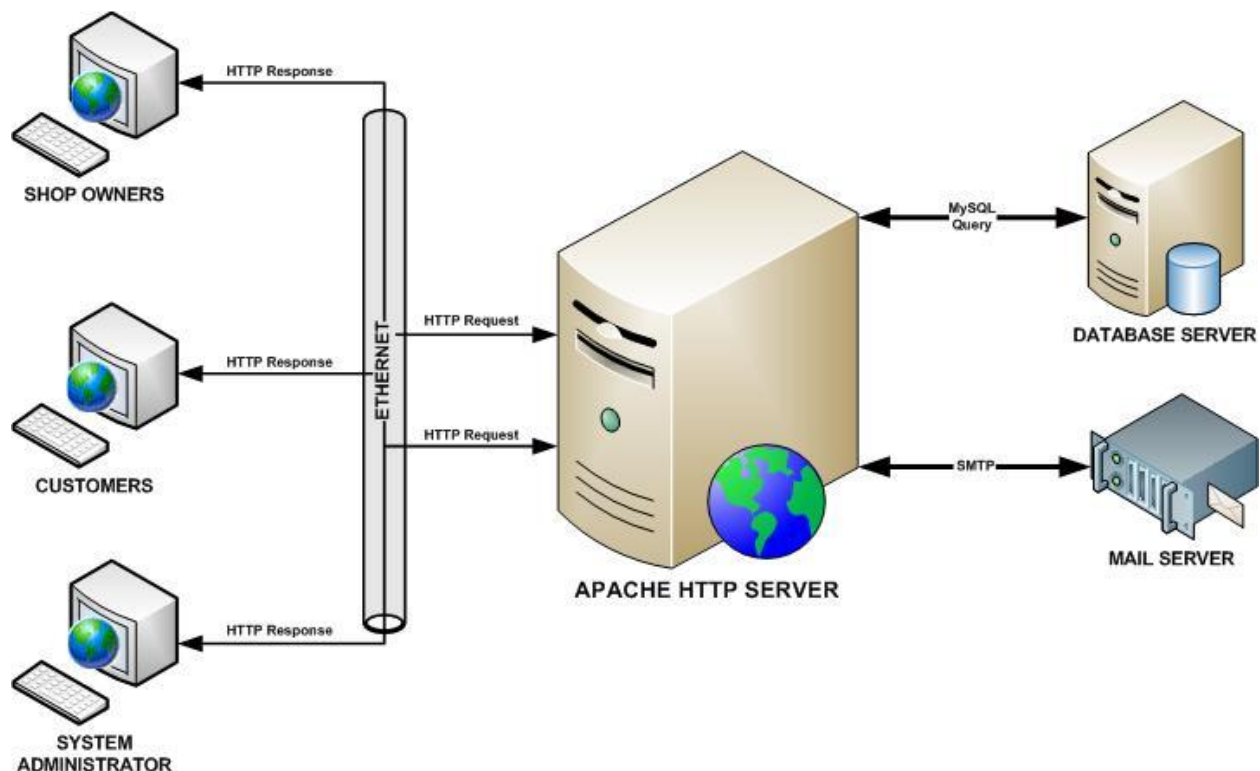


Figure 3.5: LAMP System Architecture

The system uses the concept of sessions. A session is created whenever a user logs in and that session is destroyed whenever the user leaves the browser. Because the user's session is destroyed when the user exits, cookies are used to remember preferences such as language and colour selection and also to save the sortable content layout arranged by the user through drag and drop functionality.

3.5 Shop-Owners' End

Earlier in this research, we mentioned that the proper cultural markers used in the initial re-design of the shop-owner interfaces were chosen specially by the author because of the author's experience and understanding of the targeted Xhosa culture. The following sections discuss the re-development of the virtual back-end/shop-owners' end to a culturally and linguistically localized e-Commerce platform for the rural community. This section will also explain the shop-owner interfaces used in designing the system. The interfaces are only accessible after successfully logging in from the customer's end of the system. There are currently two registered online virtual shops on the shopping e-mall; the Ngwane and the Ilingelihle Arts and Craft Groups. The interfaces displayed in the following sections will be based on the interface used for the two mentioned shops.

After designing the shop-owner interfaces, there were some questionnaires and interviews which were conducted as basic user requirements elicitation with the shop-owners. The intention of conducting the questionnaires and interviews was for collecting data about the effectiveness of cultural markers in the usability of localized interfaces. The analysis and findings from the elicitation determined what changes should be made on the shop-owner interfaces. After the software was modified based on the elicitation analysis and findings, a field experiment was conducted to further test whether the localized software actually met with the shop-owners' requirements and improved the usability. The results of the field experiment are discussed in Chapter Five.

The following are interventions that have been employed to improve the functionality and display of the shop-owner interfaces:

- ❖ In the old version of the shop-owner interfaces, the shop-owners would use two separate interfaces for viewing and modifying uploaded items. In this research, we have improved the way to display the uploaded items and modifying them in one single interface.
- ❖ Text completion functionality (autosuggestion) has been introduced across the interfaces.

- ❖ Colour adaptation functionality used to change background colours instead of being limited to using just one colour across all the interfaces.
- ❖ A navigation support tool used to assist the user on how to navigate through the system.
- ❖ A principle method of linguistically localizing the interfaces.
- ❖ Audio clues for sound interaction.

The interface screenshots which will be used to assist with the explanation of the functionalities may not be clear or visible to the reader with some being displayed in isiXhosa instead of English. The reason for displaying the screenshots with smaller font sizes is to reduce the amount of space consumed. If the screenshots are not clear or visible to the reader, we have created a user account that will allow the reader to view the shop-owner interfaces as a shop owner. The account was set to be on read-only mode, thus allowing the reader to be able to view the shop items but not modify or make any changes to the database. The reader can visit the URL (Uniform Resource Locator), to log in to the shop-owners' end as a guest in reading mode just for browsing the shop items. The credentials for logging in as a guest are as follows:

Username: guest

Password: guest

After successfully logging in, if the font size is still too small to be legible the reader can increase the font by simultaneously pressing Ctrl and + (the Ctrl button and the plus button).

3.5.1 Viewing Uploaded Shop Items

When re-developing the shop-owner interfaces there were some factors that needed to be noted. The most important factor is that the greater the number of interfaces used, the greater the chances are that the users will be confused, which will lead to more mistakes. Hence, reducing the number of shop-owner interfaces may reduce the chances of making errors. The main feature of the view uploaded items interface is the image gallery and slideshow used to display the uploaded items. The slideshow displays all the items uploaded into the shop and it allows users to easily navigate through the item list. The use of an image slideshow is to avoid displaying the items in different pages or in a very long vertical list, especially if there are a lot

of items in a shop. Figure 3.6, below, shows the interface which was used by the shop-owners to display the shop items before the re-development of the interfaces. As we mentioned in Chapter One, the shopping e-mall system uses Paypal as its payment method. Paypal supports multiple currencies such as Dollars (\$), Euros (€), Yen (¥), etc., therefore the shopping e-mall system uses Euros (€) to determine the price of all the items on sale. From Figure 3.6, the price of the item is displayed in Euros (€) and this may confuse the shop owners as they may not understand how to use the currency. The shopping e-mall system uses Euros for the following reasons:

- ❖ The PayPal payment system has currency limitations which do not support the use of the South African currency (Rands).
- ❖ The marketing focus and target of the shopping e-mall system is the global market using the Internet.
- ❖ The scope of the research carried out in the first version of the shopping e-mall system by Njeje (2007) was to find a working payment system that will accept credit card payment, so that there were no constraints on the currency to be used.

The targeted Dwesa community uses the South African currency (i.e. Rand or ZAR) to price their items and displaying the item prices in a different currency would make things difficult for them.

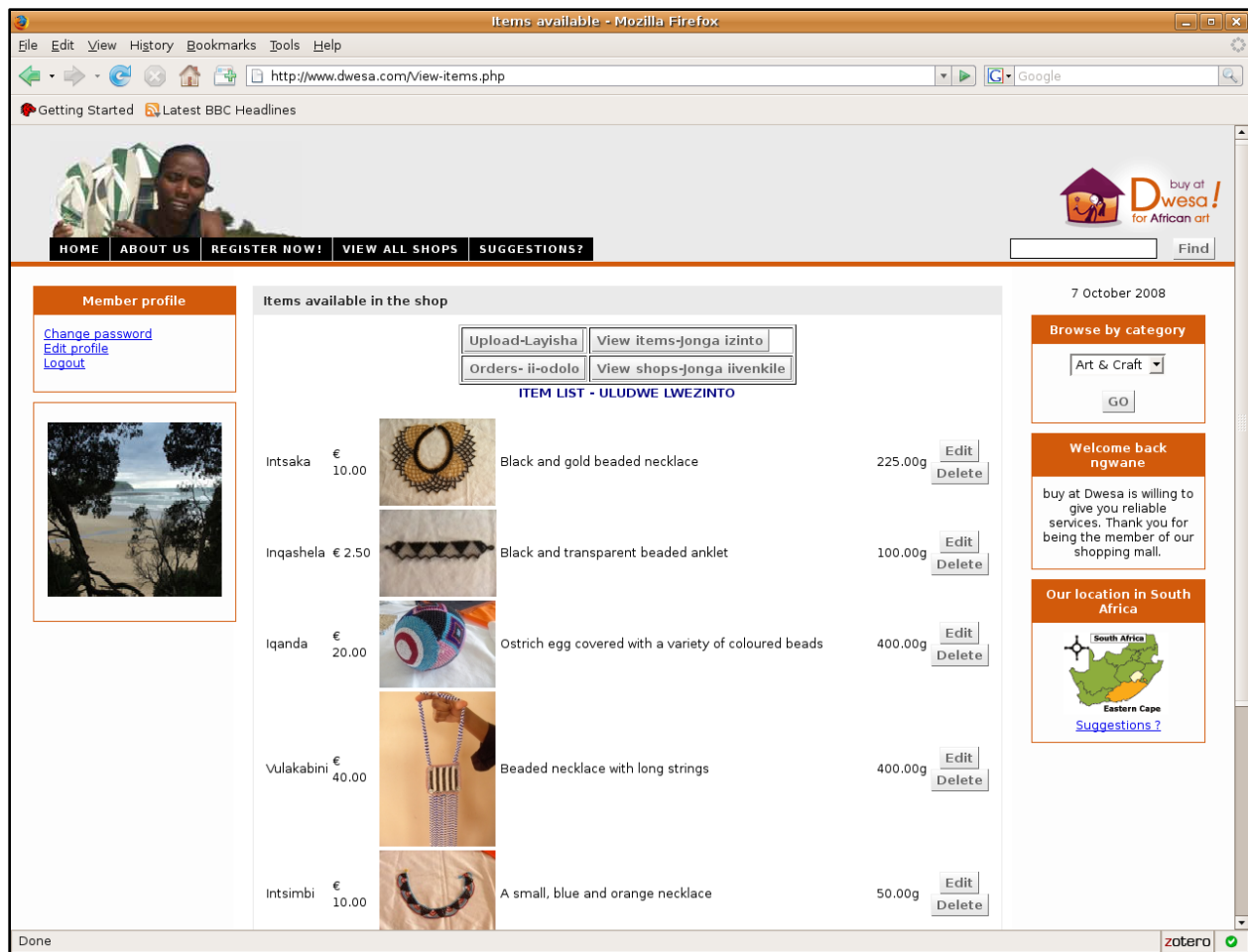


Figure 3.6: Viewing Uploaded Shop Items (Njeje, 2007)

Some elders in rural communities do not even understand the use of the term “Rand” to describe the value of money. It is a common practice that elderly people in rural communities express the amount of money in terms and words they can understand in their own language, which in this case is isiXhosa. For example, the Rand is currently trading at 12.94 to the Euro (i.e. 1 Euro $\approx$ 12.9442 ZAR $\approx$ R13) (South African Reserve Bank, 2008). When expressing R13 in isiXhosa, it is expressed as follows:

R13 = “liponti Ezintandathu Zineshumi”.

However, remembering that the some shop owners may be totally illiterate (i.e. can neither read nor write in any language) people means that expressing the value of the Rand in written words would also cause confusion. Thus, in the research one does not need to translate and express the prices in words but only convert the prices to the Rand as some of the users may

understand that currency. Not only does it apply to currency but it may also apply to weights, time and date. The following Figure 3.7 displays the localized version of the page for displaying uploaded shop items.

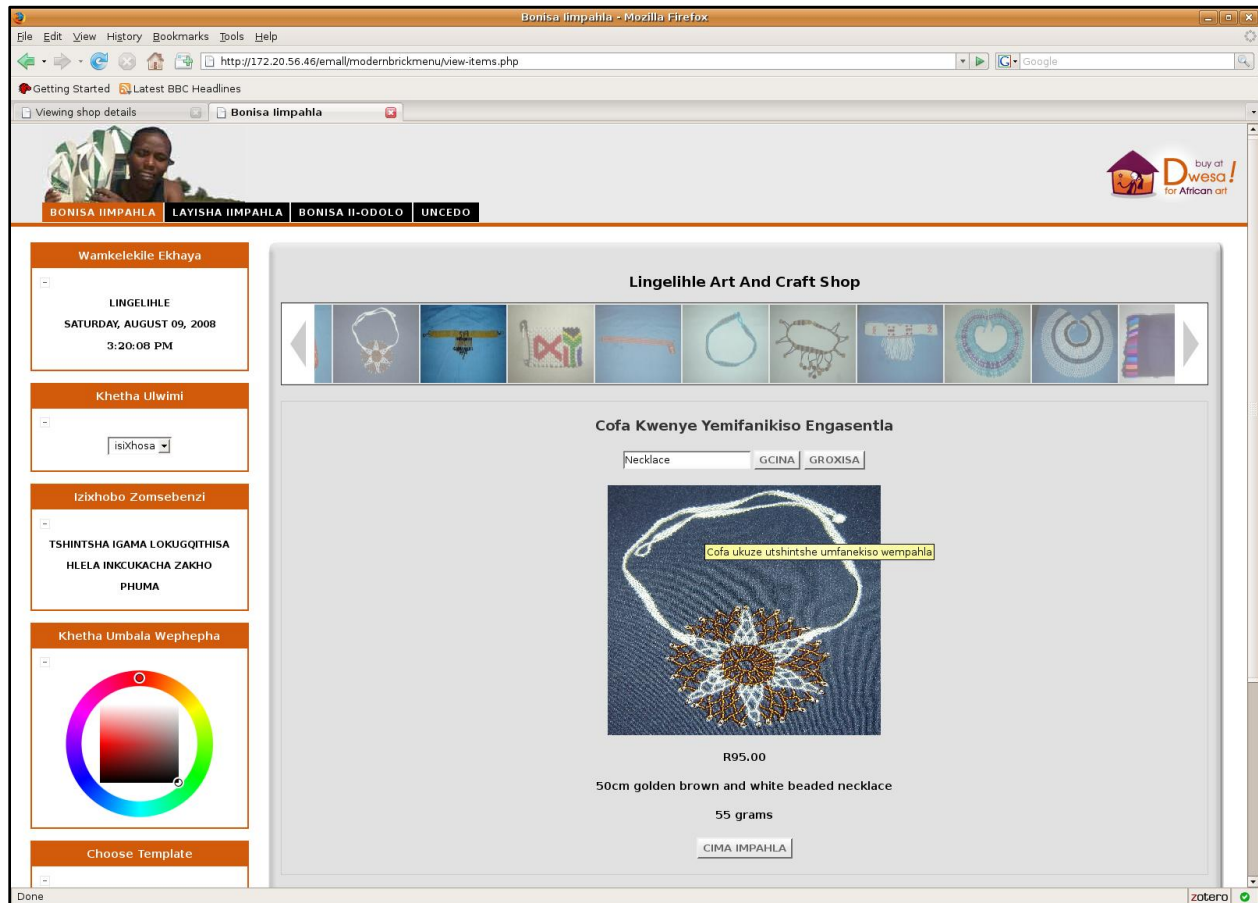


Figure 3.7: Viewing Uploaded Shop Items

Figure 3.8 displays the image slideshow used by the shop owners for browsing through the items they have uploaded into their shop. The slideshow displays all the items which are advertised and sold in the customer's end of the shopping e-mall system. The slideshow allows the user to browse from left to right and back again by using the left and right arrows of the image slideshow. Browsing in both directions is triggered whenever the user moves the mouse cursor over the directional arrows and the slideshow will browse through the direction the user has selected.



Figure 3.8: Image Slideshow

Whenever the shop owner clicks on one of the thumbnails in the slideshow, all the details referring to that selected item will be displayed below the slideshow along with the larger image of the thumbnail.

The details (name, price, image, description and weight) of the selected item are displayed in the gallery and can be edited by the shop owner from the same web page using an edit-in-place functionality. This functionality allows shop owners to replace the text and sentences they want to edit with textboxes or textareas respectively. The textboxes or textareas come with the Save/Update and Cancel buttons. The shop owner can use the textboxes to update the details which are automatically stored on the MySQL database after clicking the Save/Update button. When editing the item name details, an autocomplete and autosuggest functionality also assists the shop owner as they are inexperienced computer users who are unfamiliar with the use of the keyboard. This sort of functionality would help users complete the word faster than they could have if expected to type it, whilst avoiding spelling mistakes. When uploading an item image, there is an extra button for browsing through to the directory for the required image to replace the current image. If the shop owner replaces the existing image, the new image is resized to the correct size for thumbnails and is then stored on the image slideshow and also in the folder containing the thumbnails.

The shop owner also has an option of deleting the image from the database and from the image slideshow. Figure 3.9, displays the part of the image gallery that can be used when editing the details of the selected item.

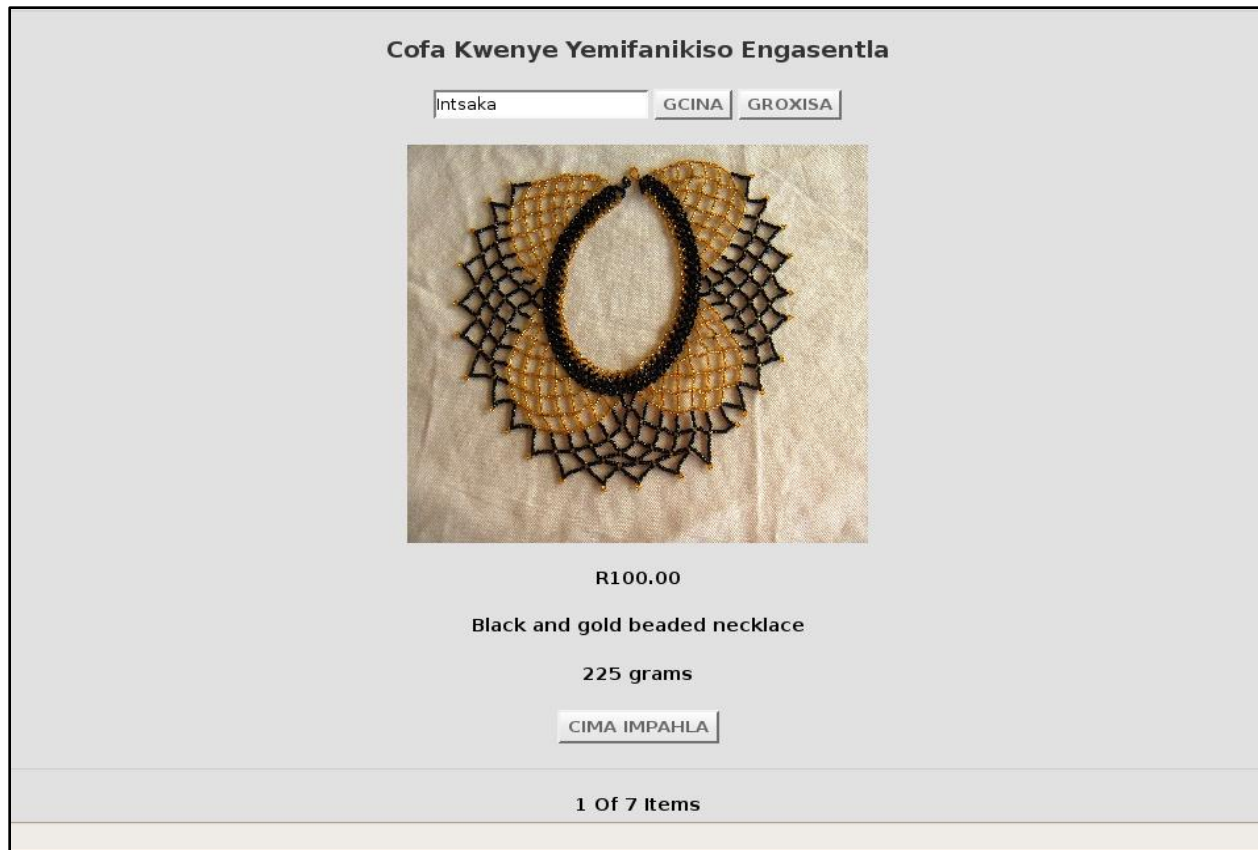


Figure 3.9: Example of the Edit in Place Functionality

3.5.2 Uploading Shop Items

The interface displayed in Figure 3.10 is used by the shop owner to upload new beaded artifacts into the shopping e-mall system. The details of the uploaded items are stored in a MySQL database table. Figure 3.10 displays the actual interface used for uploading items. The interface also provides an autosuggest functionality to help the shop owners. The autosuggestion functionality makes use of database tables to search for matching cases. For example, the autosuggestion displayed in Figure 3.10 is for the item name. The autosuggestion function searches the database for each character typed into the item name text field and displays the results in a list below the said text field. After the upload of new items, a preview of the previously uploaded item is displayed to the shop owner. From the interface with the preview, the shop owner can use the edit-in-place functionality to correct mistakes that were made during the upload. This is important because the shop owners should make sure that the details of the items they upload are correct to avoid confusion when there is an order made on their

shop. For example, if the shop owner enters an incorrect item price, the customer may end up being overcharged for an order he/she made or the seller may get less money for that product. This would be bad service on the part of the shop-owners and the system would be untrustworthy for the customers because of the bad service provided (Cyr, 2004). The following section discusses the interface used to view orders made by the customers on the customer's end.

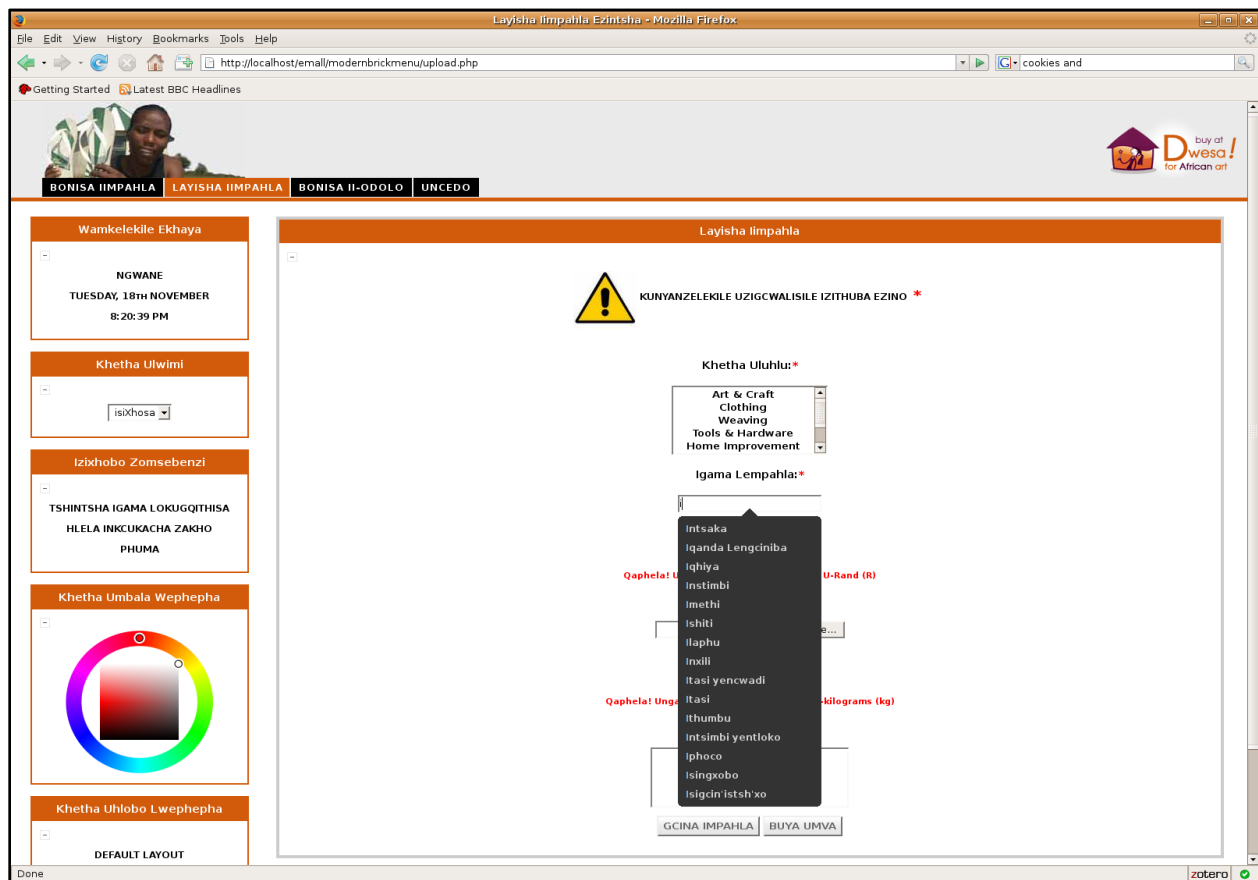


Figure 3.10: Uploading New Shop Items

3.5.3 Viewing Customer Orders

This interface is used to display orders made by customers on the customer interfaces. Whenever a customer places an order on a shop, the shop owners automatically receive an email notifying them about that order (later on SMS – Short Message Service notification). The email is sent to the email address registered by the system administrator when registering the shop. When the shop owners view the customer orders, they have an option of displaying only

delivered orders or pending orders. The aim of separating the two interfaces is to avoid confusion and at the same time keep an order history. This means an order is never deleted from the database so as to display the purchasing statistics of each customer. For example, one of the projects in progress undertaken at the Siyakhula Living Lab, is the implementation of a reward based module for the shopping e-Commerce platform deployed at Dwesa. Some of the rewards are calculated based on the purchasing statistics of a customer. Hence, there is a need to keep the order history for each customer who has made an order on the system. The following Figure 3.11 displays the interface used to display pending orders and the shop owners can also use it to display delivered orders.

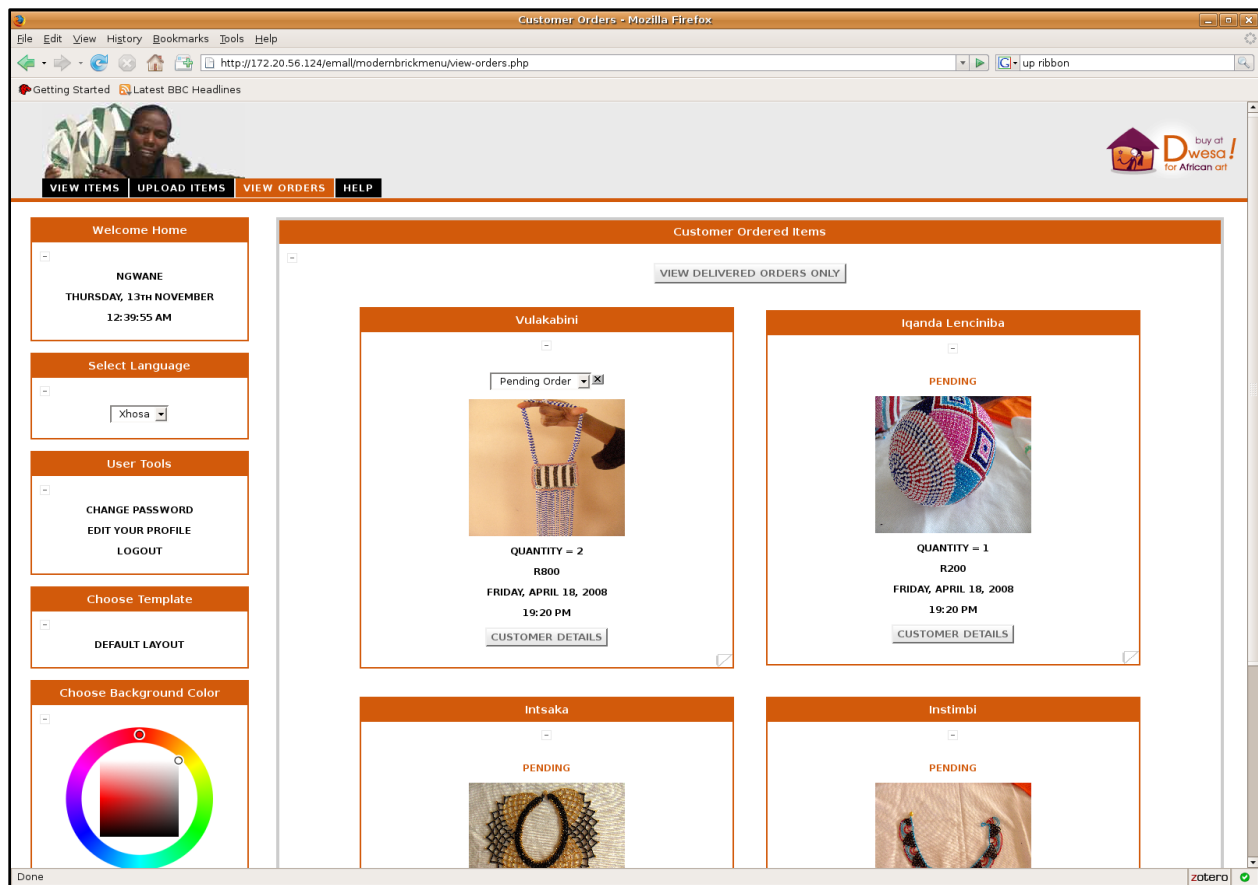


Figure 3.11: Viewing Customer Pending Orders

The displayed order details consist of the item name, image, quantity (i.e. how many orders of that item), total price, as well as date and time of the order. Also included in these details is the order status, with an edit-in-place functionality that uses a drop down option to change the order status from pending to being delivered. The status of an order is changed when the shop

owner has already delivered the order. The customer details (i.e. email address, postal address, phone number, etc.) are viewed per order display without using another interface. The interface for displaying delivered orders is similar to the interface for displaying pending orders. The only difference is the change of order status and the MySQL query used to retrieve the delivered order(s). As said earlier, the system uses automatic email notification of the customer's orders to the shop owners. In the future, it will also include automatic SMS notifications depending on the settings the shop owner chooses during registration.

3.5.4 Navigation Support Tool

The shop-owners' end of the shopping e-mall system is specifically designed for users who have a limited experience with the use of computers. Hence, it is important that the system is designed to suit their lack of understanding of how to use computer applications and services. The computer living labs installed at the five schools at Dwesa (Mpume, Mtokwane, Ngwane, Nondobo, and Nqabara) have been used for computer training purposes thus far. The author has been a trainer during the training sessions and has encountered that it was difficult for the trainees to perform their given tasks without assistance from the trainers. The elderly people would forget the steps which were taken in the previous demonstration or training sessions. This meant that the trainers had to be patient and tolerate numerous requests to repeat their past training session. Hence, the use of the navigation support tool was introduced.

This part of the shop-owner interface uses the system functionality diagram (Figure 3.3) to build the website sitemap. The sitemap is designed with the use of interface screenshots for simple user assistance and guidelines. The system sitemap is a navigational tool that provides the shop owner with guidelines that assist them in to knowing how the system operates and how to use it. The system sitemap does not only provide interface screenshots to the shop owners, but also provides collapsible and expandable popups for specific functionalities. These functionalities will be discussed in the following sections and they include language selection and a color picker. Figure 3.12, displays the screenshot of the web-page used for administrative assistance to the shop owners.

From this interface, the shop-owners can perform all the functionalities used in the shop-owners' end. The implementation of this interface will also be discussed in Chapter Four. In future, whenever the shop-owner is browsing an interface, the navigation support tool should be displayed on an open popup window. The window should indicate where the shop-owner is actually located or navigating across the tasks provided in the shop-owner interfaces.

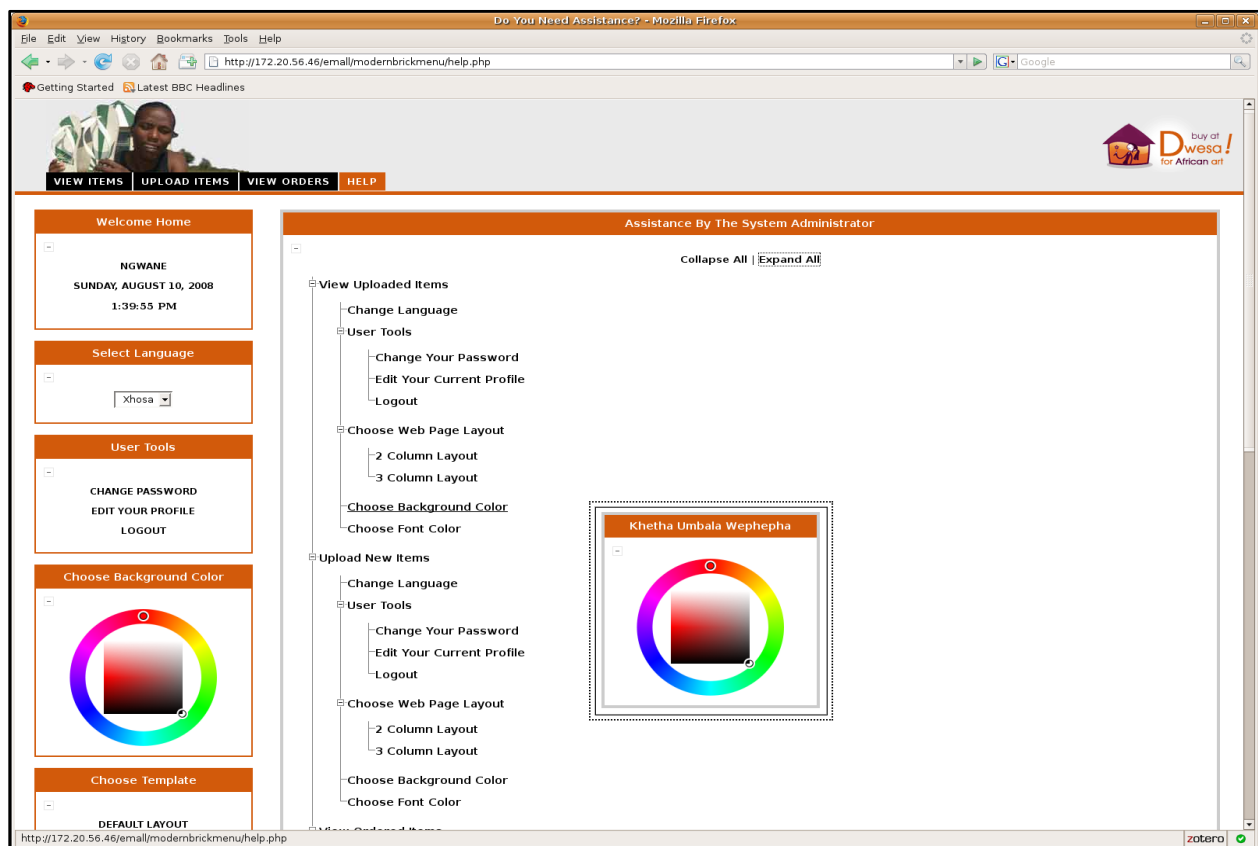


Figure 3.12: Navigation Support Tool for Shop Owner Assistance

3.5.5 Language Localization

The tools and techniques which are used for translation have been briefly discussed in Chapter Two. We also mentioned that the research makes use of the combination of GNU gettext and PO files because of the portability of PO files and the fact that less has been done, up till now, to translate into African languages such as isiXhosa. In this section, the method of translating using gettext and PO files is discussed and also how the translation is undertaken will be explained. Figure 3.13 displays the steps used in the translation process.

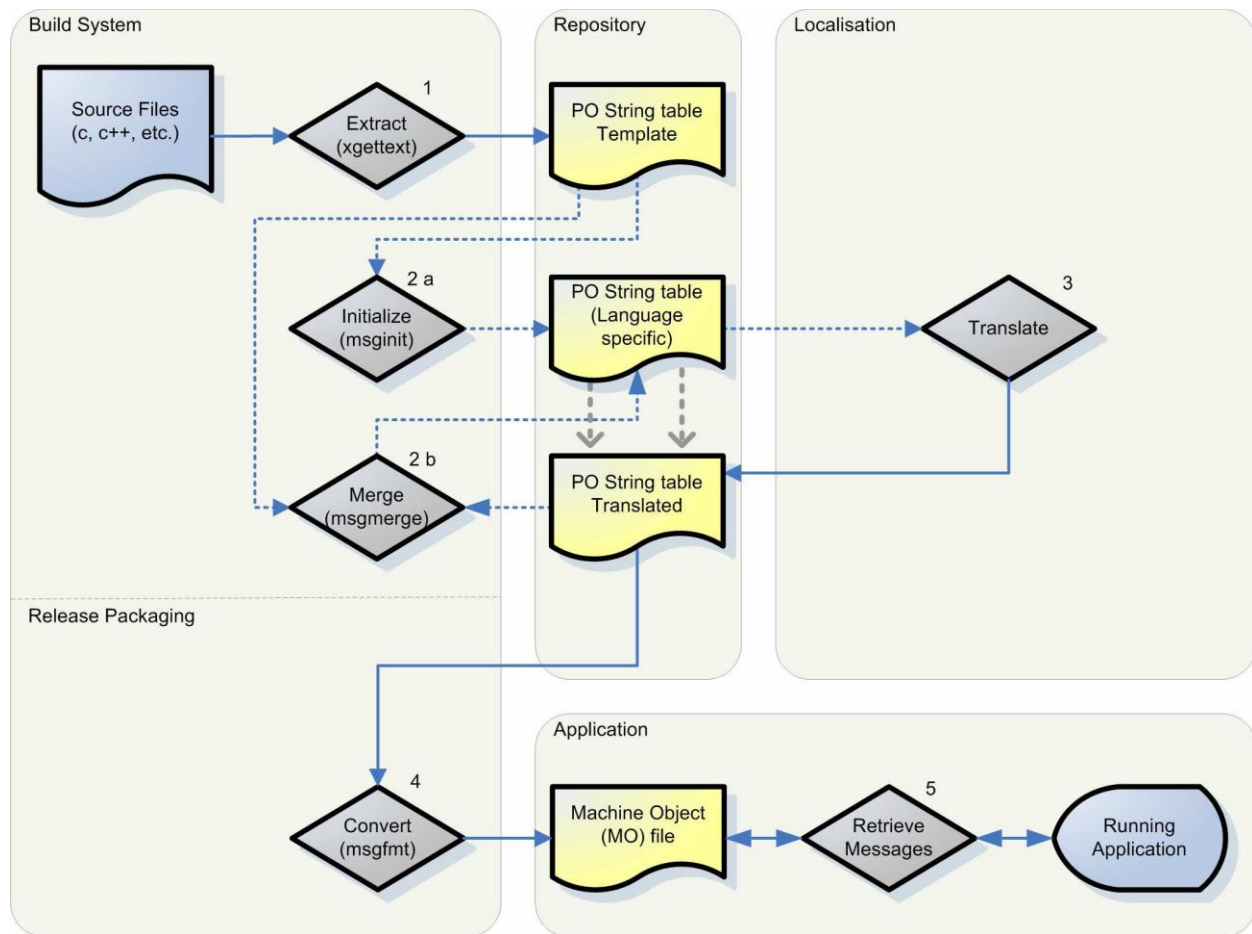


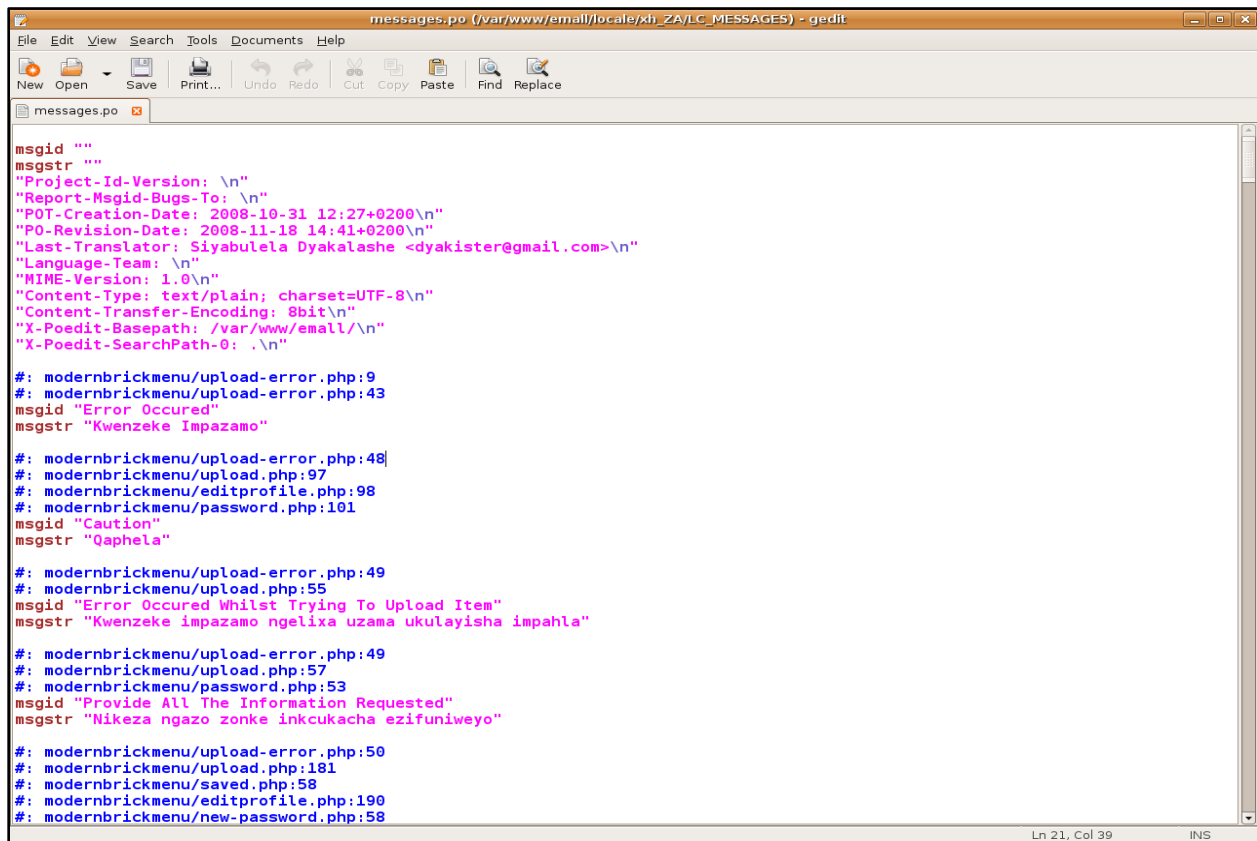
Figure 3.13: GNU Gettext Localization Process (Frimannsson, 2005)

In Chapter Two, a basic GNU gettext localization process diagram (Figure 2.1) was presented and a more detailed diagram displayed in Figure 3.13 will be discussed in this chapter. The following explanations (Frimannsson, 2005; Drepper et al., 2007) are based on the commands and steps used in the localization process:

1. Extract translatable strings from the source files into a PO template (POT) file using *xgettext*.
2. These steps (step 2a and 2b of Figure 2.1) make use of *msginit* and *msgmerge* for initializing a POT file to a PO file and merging the translated PO file with non-translated strings from extracted POT files into an updated PO file respectively. The updates include new translation units and obsolete entries (translatable strings or entries removed from the source file).
3. The human translator translates to the targeted language.
4. Convert the PO file into an MO file using *msgfmt*.

5. Retrieve the translated strings and display on runtime. This step will be discussed in the following chapter which deals with the implementation of the shopping e-mail system.

The structure of a PO file is also important to understand. Figure 3.14 displays a sample of the PO file which was opened using a common Linux text editor. The screenshot was taken during the translation of the shop-owner interfaces and it shows different entries, each entry having a relation to the non-translated string and its corresponding translation.



```
messages.po (/var/www/emails/locale/zh_ZA/LC_MESSAGES) - gedit
File Edit View Search Tools Documents Help
New Open Save Print... Undo Redo Cut Copy Paste Find Replace
messages.po
msgid ""
msgstr ""
"Project-Id-Version: \n"
"Report-Msgid-Bugs-To: \n"
"POT-Creation-Date: 2008-10-31 12:27+0200\n"
"PO-Revision-Date: 2008-11-18 14:41+0200\n"
"Last-Translator: Siyabulela Dyakalashe <dyakister@gmail.com>\n"
"Language-Team: \n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=UTF-8\n"
"Content-Transfer-Encoding: 8bit\n"
"X-Poedit-Basepath: /var/www/emails/\n"
"X-Poedit-SearchPath-0: .\n"

#: modernbrickmenu/upload-error.php:9
#: modernbrickmenu/upload-error.php:43
msgid "Error Occured"
msgstr "Kwenzeka Impazamo"

#: modernbrickmenu/upload-error.php:48
#: modernbrickmenu/upload.php:97
#: modernbrickmenu/editprofile.php:98
#: modernbrickmenu/password.php:101
msgid "Caution"
msgstr "Qaphela"

#: modernbrickmenu/upload-error.php:49
#: modernbrickmenu/upload.php:55
msgid "Error Occured Whilst Trying To Upload Item"
msgstr "Kwenzeka impazamo ngelixa uzama ukulayisha impahla"

#: modernbrickmenu/upload-error.php:49
#: modernbrickmenu/upload.php:57
#: modernbrickmenu/password.php:53
msgid "Provide All The Information Requested"
msgstr "Nikeza ngazo zonke inkcukacha ezifuniweyo"

#: modernbrickmenu/upload-error.php:50
#: modernbrickmenu/upload.php:181
#: modernbrickmenu/saved.php:58
#: modernbrickmenu/editprofile.php:190
#: modernbrickmenu/new-password.php:58
```

Figure 3.14: The Structure of a PO file

In Figure 3.14, the PO file contains a set of translation units. Each translation unit is composed of *msgid* (source) and *msgstr* (target). The *msgid* contains the original non-translated string from the source file and the *msgstr* contains the translated version of the string corresponding to the non-translated string. There is also some metadata contained within each translation unit which provides the references to the source filename and the line number from which the non-translated string in the *msgid* was retrieved (see Figure 3.14). The metadata can also

provide information about the comments by the translator or comments which were extracted from the source file.

3.5.6 Color Picker

The shop-owners' end was developed in such a way that it can be replicated in other rural marginalized communities of South Africa and successfully elsewhere in Africa. This means cultural diversity should be taken into account and supported. This introduced the study of using cultural markers (i.e. colour, images, layout, etc) in multilingual web designs (Barber & Badre, 1998; Sun, 2001). The common method used to obtain cultural markers is interviewing the target users of the system and filling in questionnaires. However, due to the circumstances mentioned earlier and the possible development of this research in other regions, the author developed a tool that would enable the shop-owner to choose which suits their cultural preference from the system. The most important factor about introducing such a tool is because colours have different meanings in different cultures. Some colours can be found offensive in certain cultures and may be deemed disrespectful if a website designer chooses to use those colours. Hence, it is important that the designer researches the difference in cultures thoroughly and learns the customs of the cultural community he/she is designing the product for.

This confusion is not only limited to colours but also symbols. For example, the requirements of a product or software designed for a rural community in South Africa may differ from that of a rural community in another African country. Therefore, when developing software for multiple cultures, it is important that the software be flexible in adaptability and not limit the design to a single specific culture. In this research, a color picker (displayed in Figure 3.15) has been implemented for making colour changes to the interface design.

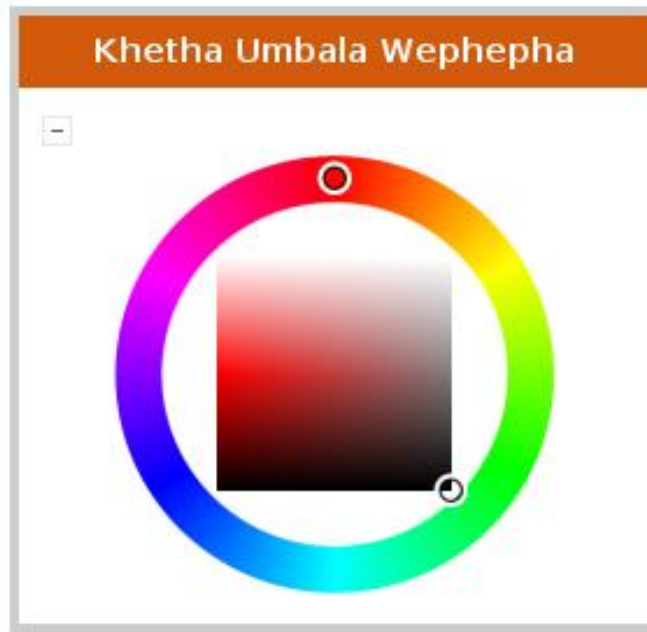


Figure 3.15: Color Picker

The implemented color picker is called the Farbtastic Color Picker (Wittens, 2007) and it is a JQuery plug-in. According to the JQuery homepage (Resig, John and the JQuery Team, 2008), *“JQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and AJAX interactions for rapid web development”*. The farbtastic color picker is developed in such a way that the user uses a dragging functionality to select a colour. The color picker has two colour markers; the outer (on the circle) and the inner (on the square) colour markers. The reason for having two colour markers provides an advanced option for the blending of different colours.

Once a colour is selected it is stored in a hidden textbox which is later read by a JavaScript function called *selectedColor*. The JavaScript function makes use of two parameters, the selected colour in hexadecimal format and an *id* which is the encryption of the username and password. This function is used to store the selected colour in a cookie so that it can be recalled again whenever the user revisits the shop-owner interfaces. Whenever the shop-owner releases the mouse after dragging, that interface is refreshed so as to display the new colour settings. The details on how the Farbtastic color picker was implemented will be discussed in the next chapter on implementation.

In addition to this kind of functionality, a drag and drop functionality has also been implemented so that the shop owners can create and use different interface layouts. The drag and drop functionality will be explained in the following chapter, Chapter Four.

3.6 Audio Clues

The implementation of a text-to-speech type of audio interaction for non-English speaking users is difficult especially when implementing it in an African language. To implement an interface audio interaction (i.e. text-to-speech and speech recognition) that fully operates in isiXhosa is complex and may also take a long time because of recording speeches, voices, and sounds. Hence, due to time constraints, audio clues have been implemented but are only limited to hyperlinks and certain functionalities used in the shop-owner interfaces. The audio clues were implemented using a JavaScript that will be explained in the implementation chapter which follows this one. The interaction is implemented using an onclick mouse effect which allows the user to play the recorded speech for that particular text. The process involves recording of speeches from the system designer and saving them in an audio format (e.g. wav, wma, mp3, etc). The audio files are stored in an audio directory. Each audio file is played whenever the onclick mouse effect is triggered by clicking on the play icon next to the link. If the user is using isiXhosa as the preferable language of use, the audio clues will be presented in that particular language.

3.7 Conclusion

This chapter has covered the research methodology and the design of the localized interfaces that have been used for the shop-owners' end of the shopping e-mall system. The aim was to design interfaces that are user-attractive, user-friendly, meaningful, and convenient for the shop owners. Upon completion, this software was tested in the form of a field exercise conducted for the targeted shop owners and based on the results of the lab exercise there might be some modifications on the interfaces and their functionalities. The following chapter presents the actual research implementation of the culturally and linguistically improved interfaces.

CHAPTER FOUR

Implementation

4.1 Introduction

This chapter discusses the implementation of the interventions that have been mentioned in Chapter Four. These interventions make use of the combination of the PHP, JavaScript, and AJAX scripting languages. Firstly, we will discuss how the connection to the database is established using MySQL queries in PHP.

4.2 Database Connection

Using PHP for connecting to a MySQL database requires the understanding of the MySQL functions or queries. The user connects to the MySQL database server through the interfaces by sending a MySQL query. The server replies by sending back the data requested from the MySQL query. The MySQL functions used to connect through to the database include (PHP MySQL Functions, 2009):

- ❖ **mysql_connect** – this function establishes or opens a connection on the database. It is used to establish the connection, the server name, database name, database username and password. If the connection cannot be established an error can be reported to the user explaining what caused the error.
- ❖ **mysql_select_db** – this function is used to provide the database name. This is necessary to implement as the database connection will not know which database to connect to, otherwise an error is returned.
- ❖ **mysql_query** – this function is used to send queries to the database server. This function makes use of functions such as INSERT, SELECT and UPDATE to send, retrieve or edit information stored in the database.
- ❖ **mysql_result** – this function stores the content of a cell retrieved from the MySQL query or result set.
- ❖ **mysql_num_rows** – this function returns the number of rows in a record set when using the MySQL query.

- ❖ **mysql_fetch_assoc** – this function returns the associative array corresponding to the fetched row. A similar function is **mysql_fetch_array** which returns an associative array, numeric array, or both.
- ❖ **mysql_close** – this function terminates the established connection.

As mentioned in the previous chapters, this research is a re-implementation of the deployed e-Commerce platform to better support culturally and linguistically localized shop-owner interfaces. This means that there have not been changes made to how the user connects to the database through the interfaces. The following code is in a file called *config.php* (for security purposes the correct *dbuser*, *dbpass*, and *database* are not provided here). This file stores only the information required to connect to the database.

```

1  <?php
2      $dbhost='localhost';
3      $dbuser = 'databaseusername';
4      $dbpass = 'databasepassword';
5      $database = 'databasename';
6  ?>

```

The following file, *opendb.php*, is used to establish a connection to the database and makes use of the information provided in the *config.php* file. The *opendb.php* file is displayed below:

```

1  <?php
2      $conn = mysql_connect($dbhost, $dbuser, $dbpass);
3      @mysql_select_db($database) or die('Error connecting to mysql');
4  ?>

```

The connection is closed using *closedb.php* which contains the *mysql_close* function. The following discusses how the language selection is implemented and how the connection to the PO files is established using PHP.

4.3 Preloading User Settings

Chapter Three discussed the interventions which were implemented in this research. These include colour adaption, language selection, the navigation support tool, sortable layouts, text completion (i.e. autocompletion and/or autosuggestion), and sound interaction. However, the primary concern was on how the changes made using these interventions could be stored so

that users do not have to redesign the interface every time they use the shop-owner interfaces. There are different ways one can store this data but in this research we have made use of the database for capturing this data or user settings.

Before the browser redirects the shop-owners to their home page, the *settings.php* script is carried out. This PHP script loads all the shop-owners' settings (i.e. all the interventions mentioned above) based on the data stored in the database. The following PHP code displays how the language settings for a specific user, identified by *shopid*, are loaded:

```
1  <?php
2      session_start();
3      $username = $_SESSION['user'];
4      include("config.php");
5      include("opendb.php");
6      $shop_query = "SELECT * FROM STORES WHERE Username='$username'";
7      $result = mysql_query($shop_query);
8      $num = mysql_num_rows($result);
9      $i = 0;
10     while($i < $num){
11         $shopid = mysql_result($result, $i, "Shopid");
12         $i++;
13     }
14     $_SESSION['shopid'] = $shopid;
15     $setting_query = "SELECT * FROM SETTINGS WHERE Shopid='$shopid'";
16     $results = mysql_query($setting_query);
17     $num_rows = mysql_num_rows($results);
18     if($num_rows == 0 and $shopid != NULL){
19         $insert_query = "INSERT INTO SETTINGS (Language) VALUES('en_ZA')";
20         mysql_query($insert_query) or die("Not executed");
21         $_SESSION['language'] = "en_ZA";
22     }
23     else{
24         while($row = mysql_fetch_assoc($results)){
25             $_SESSION['language'] = $row['Language'];
26         }
27     }
28     include("closedb.php");
29     ?>
```

The name of the database table for storing the language settings is called SETTINGS. It contains shopid and the locale of the language used by the last shop-owner who logged into the system. If the table is empty, the default settings are loaded into the database. Otherwise the old settings are retrieved from the database table and stored in a session which is later checked to

certify that it was set correctly. When all the user's settings are loaded on the interface, the script redirects to the shop-owners home page (*view-items.php*).

4.4 Language Selection

In Chapter Three, the method used for language localization was discussed but not the method in which it operates in PHP. The following functions are used to connect to the selected PO file (Drepper et al., 2007):

- ❖ **putenv** – this function sets the environment variable value.
- ❖ **setlocale** – this function specifies which locale to use based on the selection. For example, in this research the locale for English and isiXhosa is en_ZA and xh_ZA respectively. This function is initialized by using the following functions and locale categories:
 - LC_ALL – this environment variable determines the value for all categories
 - LC_MESSAGES – this variable determines the locale category for system responses
 - LC_TIME – locale category for formatting date and time information
 - LC_NUMERIC – defines symbols for formatting non-monetary numeric information
 - LC_COLLATE – determines the locale category for character and string collation information
 - LC_MONETARY – defines symbols for formatting monetary numeric information
 - LC_CTYPE – for character case conversion and other character handling functions (e.g *tolower()* and *toupper()*)
- ❖ **bindtextdomain** – provides the domain (i.e. the name of the PO file) and directory where the gettext locale folders are stored.
- ❖ **textdomain** – this function provides the domain to always use.
- ❖ **gettext** – this function identifies which strings or sentences to be translated. When using gettext with PHP to get the translated string we have to make use of the *gettext()* function. This function can also be replaced by *_()* which is just a short way of writing *gettext()* all the time. For example:
echo gettext("Hello"); or echo _("Hello");

Creating locale directory structures is also very important to understand. In this research, the following locale directories are used:

/locale/xh_ZA/LC_MESSAGES/messages.po and /locale/xh_ZA/LC_MESSAGES/messages.mo

/locale/en_ZA/LC_MESSAGES/messages.po and /locale/en_ZA/LC_MESSAGES/messages.mo

As mentioned in the previous chapters, each PO file supported should have its own locale directory. To extract translatable strings without using some of the tools mentioned in Chapter Two, the *xgettext* command is used on the command terminal. This command scans all PHP files in the project directory searching for references to *gettext* and this command generates a PO file named *messages.po*. The command is executed as follows:

```
siyabulela@siyabulela-desktop:/var/www/emall$ xgettext -n *.php
```

After the PO file has been generated it needs to be compiled and output to a binary *messages.mo* file. This is executed as follows:

```
siyabulela@siyabulela-desktop:/var/www/emall$ msgfmt message.po
```

The *gettext* function only works for locales that are recognized by the Linux operation system. This means that the en_ZA and xh_ZA locales will not be recognized and the translation will not work at this point. For the locales to be recognized, the designer should make use of the *locale-gen* command. The *locale-gen* command is executed as follows and if the command was executed successfully, a report is sent to the user:

```
root@siyabulela-desktop:/var/www/emall/locale# locale-gen xh_ZA
Generating locales...
  xh_ZA.ISO-8859-1... done
Generation complete.
root@siyabulela-desktop:/var/www/emall/locale# locale-gen en_ZA
Generating locales...
  en_ZA.ISO-8859-1... done
Generation complete.
```

The *localization.php* file stores information about which locale to select when a user selects a language. This file is called in all shop-owner interfaces using the *require_once* function i.e. *require_once("localization.php")*.

```

1  <?php
2      if(isset($_SESSION["language"])){
3          include("config.php");
4          include("opendb.php");
5          $query = "SELECT Language FROM SETTINGS WHERE Shopid='$shopid'";
6          $result = mysql_query($query);
7          while($row = mysql_fetch_assoc($result)){
8              $locale = $row['Language'];
9          }
10         include("closedb.php");
11     }
12     $gettext_domain = 'messages';
13     putenv("LC_MESSAGES=$locale");
14     setlocale(LC_ALL, $locale);
15     bindtextdomain($gettext_domain, './locale');
16     textdomain($gettext_domain);
17  ?>

```

From this script, we can see that the locale is always read from the database instead of a cookie or session. When a user selects a language, the locale settings for that language are stored in a database table that stores two entries, the *shopid* and locale. Software designers often make use of cookies and sessions for storing this type of data, but the problem with cookies and sessions is that they expire and are stored on the web browser. This means, if the user decides to use a different computer, the user's previous settings will not be traceable or loaded on that computer's web browser. Hence, the user would have to redo all the settings he/she had done in the past. Our aim was to store the user's settings in such a way that all settings are loaded regardless of which computer or browser the user is utilizing. Each user that visits the shop-owners' end will be able to use isiXhosa as the default language and they can always change from that language to English. There are numerous alternative ways one can use for storing the user's setting. These include the storage of configuration items with INI files, XML configuration files, PHP, text files, or a database. In this research we chose to make use of the database to store the user's settings. The reason for using the database to store the user's settings was because we wanted the user's settings to be loaded on any computer they might use.

To store the selected language locale in the database, we used the *shopid* to identify which user the settings should apply to. The reason for using the *shopid* is to separate the settings of different users who make use of the shop-owner interfaces. The *shopid* and the *locale* are

passed through to a JavaScript function called *lang_selected* which take these as its parameter. The shop-owners use a drop down option for language selection and this triggers the JavaScript function. The following code shows how the selected *locale* and the *shopid* are passed through to the JavaScript function to store the language settings in the database.

```
35 <select class="locale" size="1">
36     <option id="xh_ZA" onclick="lang_selected(this.value, '<?php echo($id);?>');" value="xh_ZA">
37         <?php echo(_("Xhosa"));?>
38     </option>
39     <option id="en_ZA" onclick="lang_selected(this.value, '<?php echo($id);?>');" value="en_ZA">
40         <?php echo(_("English"));?>
41     </option>
42 </select>
```

Because JavaScript is a client-side scripting language, we cannot use it to process queries to the database thus AJAX was used to call a PHP script that can process the database queries. The following JavaScript code is used for updating the database with the language the user prefers to use:

```
= function lang_selected(locale, shopid) {
    var remoteCartObj = createAjaxRequest();
    if (remoteCartObj)
    {
        remoteCartObj.open("GET", "update_language.php?shopid="+ shopid +
            "&locale="+ locale + "&rand=" + new Date().getTime(), true);

        remoteCartObj.onreadystatechange = function()
        {
            if (remoteCartObj.readyState == 4 || remoteCartObj.readyState == 0)
            {
                if (remoteCartObj.status == 200 || remoteCartObj.status == 304)
                {
                    window.location.reload();
                }
            }
        };

        remoteCartObj.send(null);

        return false;
    }
}
```

As we can see on the JavaScript code, the AJAX object called *remoteCartObj* opens the PHP script called *update_language.php* and also parses the *shopid* and *locale* to the script. The *update_language.php* script is as follows:

```
1  <?php
2      sleep(1);
3      include("config.php");
4      include("opendb.php");
5      $shopid = $_GET['shopid'];
6      $locale = $_GET['locale'];
7      $query = "UPDATE SETTINGS SET Language='$locale' WHERE Shopid='$shopid'";
8      mysql_query($query);
9      include("closedb.php");
10  ?>
```

After the query has been executed and the locale stored successfully, the browser is refreshed so that the language changes can take place. This means *localization.php* is executed and it replaces the old locale with the current locale stored in the database. Whenever the shop-owner revisits the shop-owners' end, the locale stored in the database is read using the *shopid*. This approach of changing the user's settings is similarly implemented also for colour changing and sortable layouts.

4.5 Background Colour Selection

In Chapter Three, we mentioned that the farbtastic color picker used for background colour selection is implemented using the JQuery plugin and JavaScript. The colour picker displayed in Figure 3.15 was implemented using the div element displayed below:

```
80  <div class="itemContent">
81      <div id="picker" align="center"></div>
82  </div>
```

The *div id="picker"* element is used to display the colour picker and for the colour picker to work we have to decide which id to use for it. This means that all interfaces that make use of the colour should include the following JavaScript code that makes use of the JQuery plugin:

```

33     $(document).ready(function() {
34         $('#picker').farbtastic('#color');
35     });

```

The JQuery version being used in implementing the colour picker and other interventions in this research is the latest version 1.2.6. The farbtastic colour picker package comes out with a farbtastic JavaScript and CSS which displays the PNG files used by the colour picker. These should also be included in all interfaces intending to use the farbtastic color picker. For the colour picker to change the background colour, the following should be included in the body tag:

class="colorpicker" style="background-color:<?php echo(\$selected_color);?>";

The background colour changes every time the user drags the mouse to a certain colour on the Farbtastic color picker. In a similar manner to how the language selection locale is stored in the database, the selected colour is stored using a JavaScript function named *released*. The JavaScript function is triggered when the user stops dragging and then releases the mouse. The function uses the AJAX object mentioned in section 4.4 and parses the last dragged colour before release and the shopid to the PHP script, *update_color.php*, for updating the colour stored in the database. *update_color.php* functions similar to *update_language.php* because the main difference is related to the parameters they receive, colour and locale.

The colour is retrieved from a hidden textbox that changes its value every time the colour picker is dragged to a certain position or colour of the farbtastic color picker. When the shop-owner stops dragging to his/her colour of choice, the JavaScript function, *released*, is triggered. After the *released* function has been successfully executed, the interface refreshes so that the changes made to the background colour can be displayed. The JavaScript function is as follows:

```

function released(){
    var color = document.getElementById("color").value;
    var shopid = document.getElementById("shopid").value;
    var remoteCartObj = createAjaxRequest();
    if (remoteCartObj)
    {
        remoteCartObj.open("GET", "update_color.php?shopid="+ shopid +
            "&color="+ color + "&rand=" + new Date().getTime(), true);

        remoteCartObj.onreadystatechange = function()
        {
            if (remoteCartObj.readyState == 4 || remoteCartObj.readyState == 0)
            {
                if (remoteCartObj.status == 200 || remoteCartObj.status == 304)
                {
                    window.location.reload();
                    //document.bgColor = color;
                }
            }
        };

        remoteCartObj.send(null);

        return false;
    }
}

```

The hidden textbox field is used to store the selected colour which is then read by *released* on mouse release (*onmouseup*) function. *id="color"* is used in the farbtastic JavaScript piece of script mentioned earlier above, *\$(‘picker’).farbtastic(‘color’)*. This piece of PHP code is included in all interfaces so that the colour settings (whether the cookie has been created or not) can be recalled. This means the colour settings stored on one interface can be retrieved on all other interfaces. This applies also to the cookie used to store the language settings.

4.6 Image Gallery and Slideshow

The discussion of the image gallery and slideshow functionality will be divided into the following two subsections; namely, Image Slideshow and Image Gallery.

4.6.1 Image Slideshow

The image slideshow uses the MySQL database to retrieve the directory URL for the item images that have been uploaded into the shopping e-mall system. The image slideshow makes

use of two files named *gallery.css* and *image-slideshow.js*. With the JavaScript file, the image slideshow is then initialized. Also in the *body* tag, the developer should include that on load the slideshow should be initialized. This means the use of *onload="initSlideShow()"* is very important as it calls the JavaScript function for initializing and triggering the slideshow. This code shows how the item images are uploaded into the image slideshow.

```

77 <div id="thumbnail_container">
78     <div id="arrow_left"></div>
79     <div id="arrow_right"></div>
80     <div id="theImages">
81         </div>
82 </div>

```

This will only load the frame for the image slideshow and not the item images till this PHP code is placed inside the *div id="theImages"* element.

```

81 <?php
82     $item_query="SELECT * FROM ITEMS WHERE Shopid='{$shopid}' ORDER by Category";
83     $result= mysql_query($item_query);
84     $numrows= mysql_num_rows($result);
85     $k =1;
86     while ($row = mysql_fetch_assoc($result)) {
87         ?>

```

This code will only fetch the URL for the directory that stores the thumbnails. As explained in Chapter Three, the thumbnails should be clickable so that they can display the details of that selected item. This onclick type of functionality is implemented as follows:

```

88 <a id="gallery<?php echo($k);?>" href="javascript:void(showPreview(
89     '<?php echo($row['Itemid']);?>',
90     '<?php echo($row['Shopid']);?>',
91     '<?php echo($row['ItemName']);?>',
92     '<?php echo($row['Price']*13);?>',
93     '<?php echo($row['Description']);?>',
94     '<?php echo($row['Mass']*1);?>',
95     '<?php echo($row['Image']);?>',
96     '<?php echo($k);?>',
97     '<?php echo($numrows);?>'))">
98     <img id="image<?php echo($k);?>" name="slide<?php echo($k);?>"
99     src="../<?php echo($row['Image']);?>" title="<?php echo($row['ItemName']);?>" />
100 </a>

```

After this has been implemented, the image slideshow will have thumbnails of the item images stored in a folder. On clicking the item image, the larger image of that item is displayed in the gallery along with the details of that item (name, price, weight and description). The item price is stored in Euros (€) in the database thus when the item price is read from the database it is converted into Rand by multiplying it by the current exchange rate value (i.e. $1 \text{ €} \approx \text{R}13$) (South African Reserve Bank, 2008). Currently, the exchange rate is statically or manually updated from the system scripts. Since the scope of this research does not cover the investigation of a tool that would allow the exchange rate to be automatically updated throughout the system, it is open for exploration in future. The following subsection explains how the item details are displayed in the image gallery.

4.6.2 Image Gallery

The item details are displayed using a JavaScript function called *showPreview* which contains the variables for itemid, shopid, item name, price, description, mass, image directory, a counter and the total number of items in the shop. The *showPreview* function also displays these details as shown in Figure 3.8. As said in Chapter Three, the image gallery is supposed to support edit-in-place functionality which uses a JavaScript. The following code was used to display the item name once the *showPreview* function is loaded. The *showPreview* JavaScript function is implemented as follows:

```

function showPreview(itemid, shopid, itemName, price, descrpt, mass, imagePath, count, total){
    closeEdit();closeEditImage();closeEditDescrpt();closeEditMass();closeEditPrice();
    var filename = imagePath.substr(imagePath.lastIndexOf("/") +1, imagePath.lastIndexOf("."));
    var url = "thumbnails/"+shopid+"/"+filename;
    document.getElementById("display_thumbnail").innerHTML= "";
    document.getElementById("display_id").setAttribute('value',itemid);
    document.getElementById("display_shopid").setAttribute('value',shopid);
    document.getElementById("display_itemname").innerHTML= "+itemName";
    document.getElementById("display_price").innerHTML= "R"+price;
    document.getElementById("display_description").innerHTML= "+descrpt";
    document.getElementById("display_mass").innerHTML= +mass+" grams";
    document.getElementById("edit_name").value = itemName;
    document.getElementById("edit_price").value = price;
    document.getElementById("edit_descrpt").value = descrpt;
    document.getElementById("edit_mass").value = mass;
    document.getElementById("count").value = count;
    document.getElementById("numItems").innerHTML= count+" Of "+total+" Items";
    document.getElementById("delete_button").removeAttribute("style");
    document.getElementById("delete").removeAttribute("style");
    document.getElementById("delete").setAttribute('onclick','javascript:deleteItem('+itemid+')');
}

```

When the shop-owner clicks on the item name it is substituted with a textbox containing a Save and Cancel button using a JavaScript function called *edit_name*. Editing the item image, price, description and weight is done in a similar manner but via calling a different JavaScript function.

```

103 <div id="textbox_name" style="display:none;">
104     <img id="loading" style="display:none;" />
105     <p id="wait" style="display:none;">
106         <?php echo(_("Please Wait. Saving ...."));?>
107     </p>
108     <div id="button">
109         <input type="text" id="edit_name" />
110         <input id="updateButton" class="update_button"
111             type="button" onclick="saveEdit();" value="<?php echo(_('Save'));?>" />
112         <input id="cancelButton" class="update_button"
113             type="button" onclick="closeEdit();" value="<?php echo(_('Cancel'));?>" />
114     </div>
115 </div>

```

The *saveEdit* JavaScript function is triggered when the shop-owner clicks on the Save button and connects to the MySQL database to store the changes. The function makes use of the combination of AJAX and JavaScript to perform this functionality. Only one textbox can be opened when editing the details on the item during the process of using the edit-in-place functionality. However, because JavaScript is a client-side scripting language the *edit_name*

function also makes use of a PHP script to execute the MySQL query required for updating the data in the database. The *saveEdit* function parses the itemid and the new item name to be stored in the database to the PHP script, *upload_name.php*, using AJAX and JavaScript.

```
2  <?php
3      sleep(2);
4      include("config.php");
5      include("opendb.php");
6      $itemid = $_GET['itemid'];
7      $itemname = $_GET['itemname'];
8      $query="UPDATE ITEMS SET ItemName='$itemname' WHERE Itemid='$itemid'";
9      mysql_query($query) or die('Error, inserting details failed');
10     include("closedb.php");
11  ?>
```

The edit-in-place functionality used for editing the item name is replicated in a similar way to the other item details. The following JavaScript code displays the *edit_name* function which replaces the clicked text with the textbox:

```
function edit_name(){
    closeAllOpenElements();
    document.getElementById("display_itemname").style.display= "none";
    document.getElementById("textbox_name").removeAttribute("style");
}
```

When the cancel button is clicked, the *closeEdit* function is triggered. This function is the reverse of what is done in the *edit_name* function. This means the textbox is hidden and the item name is displayed.

4.7 Autocompletion and Autosuggestion

The autosuggest functionality is implemented on the PHP script for uploading item details and uses the MySQL database for searching. Autosuggestion is implemented on the item name text field and searches the database for every character entered in the text field. An example of the autosuggestion is displayed in Figure 3.9. This functionality makes use of Ajax and JavaScript to display the results obtained from using a MySQL query. The autosuggest functionality makes

use of the *autosuggest.js* JavaScript file. However, for this functionality to work the following JavaScript should be included on the interface for uploading new items.

```
187     <script type="text/javascript">
188         var options = {
189             script:"autosuggest.php?json=true&",
190             var_itemname:"itemname",
191             json:true,
192         };
193         var as_json = new AutoSuggest('name', options);
194     </script>
```

On this JavaScript, we make use of *name* as the *id* element used to display the autosuggestion (i.e. `<input type="text" name="name" id="name" value="" />`). The *autosuggest.php* PHP script is used to search the database each time the shop-owner enters a character in the item name text field while it also retrieves results for that search. The results are displayed in a list that the shop-owner can use to select the item name they want to use. The results retrieved from the database are stored in an array of item names and when the query has been finished, this array is compiled to an array list of all the item names that match the search. This is done using the following piece of code which is stored *autosuggest.php*.

```
2  include("config.php");
3  include("opendb.php");
4      $query = "SELECT * FROM ITEMS";
5      $itemname_results = mysql_query($query) or die('Error, Inserting item failed');
6      while($row = mysql_fetch_array($itemname_results)) {
7          $itemname_array[] = $row["ItemName"];
8      }
9      $itemname_list = array_unique($itemname_array);
10 include("closedb.php");
```

4.8 Drag and Drop Sortables

The drag and drop functionality was implemented so that the shop-owners are able to sort the interface content layout in a way that is understandable to them. The drag and drop functionality allows the shop-owners to change the content layout of the interfaces by dragging and dropping the div elements in different positions of the interfaces. As mentioned earlier in this chapter, each registered user who makes use of the shop-owner interfaces can have the

background colour settings and language settings of his/her own choice. These settings are loaded from the database every time the user visits the shop-owners' end. For example, if the user selects a blue background colour before logging off, whenever he/she revisits the shop-owner interfaces the blue colour will be used for the interfaces background. This applies to the language settings and also to the sortable content layout settings which are stored in the database.

The following PHP piece of code shows how the sortable *div* contents were implemented in the shop-owner interfaces for drag and drop:

```

12 <div id="sidebar" class="groupWrapper">
13     <?php
14         include("config.php");
15         include("opendb.php");
16         $sql = mysql_query("SELECT * FROM SORTABLE");
17         $sort_order = array();
18         $i = 0;
19         while ($row = mysql_fetch_array($sql)) {
20             $sort_order[$i] = $row['orderID'];
21             $i++;
22         }
23         include("closedb.php");
24         for($k=0;$k<$i;$k++){
25     ?>
26     <div id="sort_<?php echo($sort_order[$k] + 1);?>" class="sortableDiv">
27         <?php
28             if($sort_order[$k] + 1 == 1)
29                 include("time.php");
30             if($sort_order[$k] + 1 == 2)
31                 include("locales.php");
32             if($sort_order[$k] + 1 == 3)
33                 include("usertools.php");
34             if($sort_order[$k] + 1 == 4)
35                 include("themes.php");
36             if($sort_order[$k] + 1 == 5)
37                 include("color_background.php");
38             if($sort_order[$k] + 1 == 6)
39                 include("color_font.php");
40         ?>
41     </div>
42     <?php }?>
43 </div>

```

In this research, the sortable *div* elements were implemented in such a way that they can be dragged and dropped within the *id="sidebar"* and the *class="groupWrapper"*. Only those *div*

elements with *class="sortableDiv"* can be sortable. From the code above, we have used an array to store the sortable order retrieved from the database. The *div* elements are then rearranged using the array order and the *"if"* statements make sure that the *div* content is displayed in the correct *div* according to its order. The storing of the sortable order is done by a JQuery and a PHP script which will both be discussed later.

Figure 4.1, below, displays the interface for viewing uploaded items into the shopping e-mall.

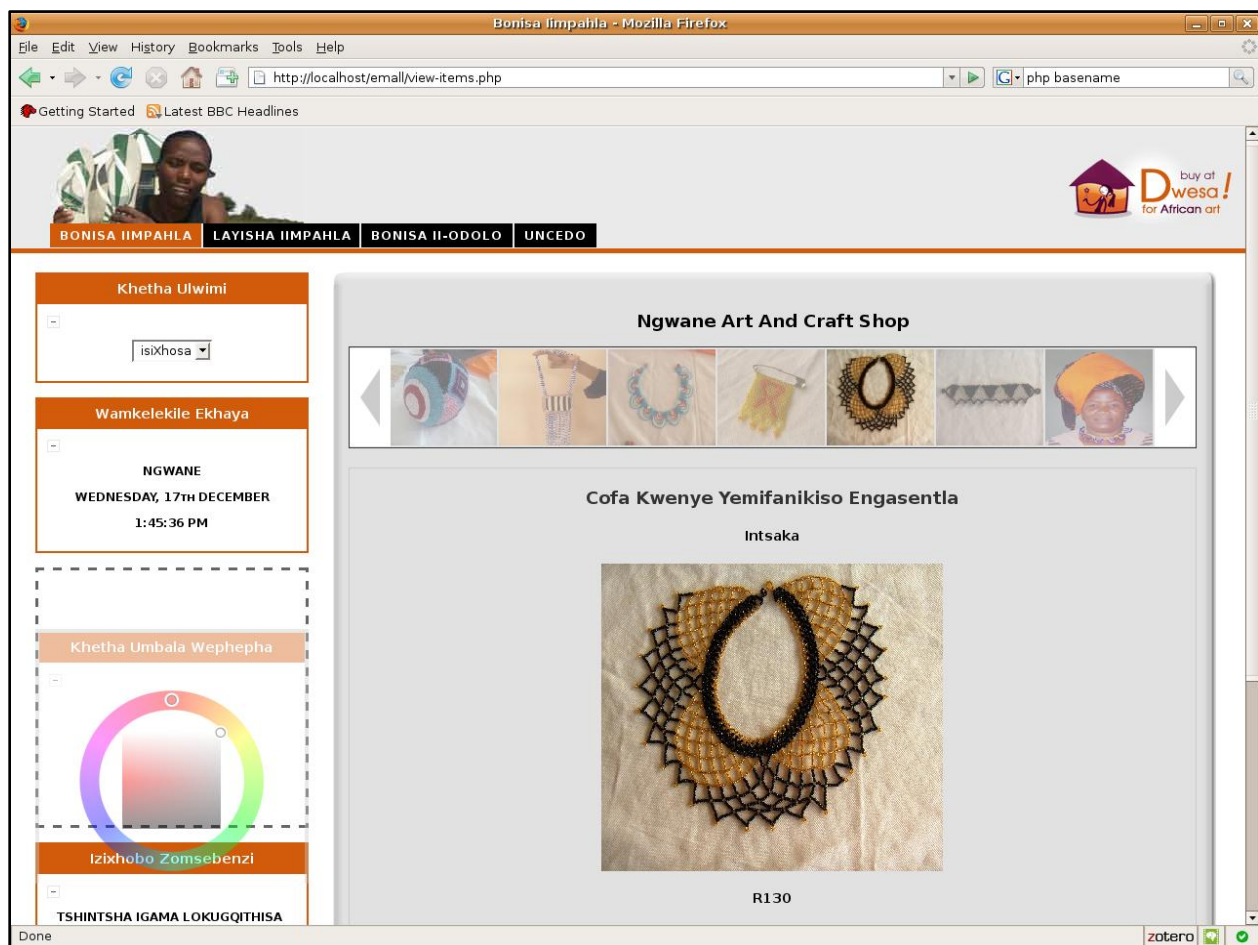


Figure 4.1: Sortable Div Elements

In the figure we see an example of how the drag and drop sortables functionality was implemented. The drag and drop sortables functionality was implemented using a JavaScript which makes use of the JQuery plugin. This functionality is not only implemented in the view items interface but in all the other shop-owner interfaces. If the shop-owner arranges the sortable *div* elements in one interface, that arrangement will be read the same way in all other

shop-owner interfaces. To store the drag and drop sortable settings in the cookie, the JavaScript uses a sequence of numbers. Each sortable *div* element used in the interfaces is assigned a number by the JavaScript so that when the cookie is read, the arrangement of the sortable *div* elements are arranged according to that sequence of numbers.

The following JavaScript is used to enable sortable *div* elements:

```
$(document).ready(function () {
    $('#a.closeEl').bind('click', toggleContent);
    $('#div.groupWrapper').Sortable({
        accept: 'sortableDiv',
        helperclass: 'sortHelper',
        activeclass: 'sortableactive',
        hoverclass: 'sortablehover',
        handle: 'div.itemHeader',
        tolerance: 'pointer',
        axis: 'vertically',
        revert: true,
        opacity: 0.4,
        onChange: function (ser) {
            // ...
        },
    });
});
```

This functionality allows the shop-owners to sort the *div* elements vertically and shows the user an area where the user can drop the dragged *div* element. For horizontal dragging of the *div* elements we can change the *axis*: *'horizontally'*. The sequence of *div* elements is stored using an *update* JQuery function with *serialize*. This JavaScript function makes use of the PHP script, *process_sortable.php*, and parses the sortable order as a parameter to it. The *update* JQuery function is as follows:

```
update : function () {
    var order = $('#div.groupWrapper').Sortable('serialize');
    $('#info').load("process_sortable.php?" + order);
}
```

The PHP script, *process_sortable.php*, is triggered whenever the user drags and drops a *div* sortable element. Its purpose is to store the sequence or order in the database. Here is a sample of the script:

```
14 |
15 include("config.php");
16 include("opendb.php");
17 foreach ($_GET['sort'] as $position => $item) :
18     $sql = "UPDATE SORTABLE SET orderID = $position WHERE sortableID = $item";
19     mysql_query($sql) or die("Not Executed");
20 endforeach;
21 include("closedb.php");
22 ?>
```

Whenever the shop-owner logs off from the shop-owner's end, the sortable sequence is not lost so that the layout sequence can be retrieved from the database when the user revisits the shop-owner interfaces using any computer.

4.9 Audio Interaction

The audio clues were implemented especially for the hyperlink but also for other functionalities. The audio clues were implemented using a JavaScript function called *soundPlay* which takes a parameter for the audio file that needs to be played. When the shop-owner points the mouse on top of the link or text to play, an icon for playing and stopping the audio is displayed. The shop-owner clicks on the play icon and the following JavaScript is triggered to play the audio file:

```
var sound;

function soundPlay(fName) {
    soundStop();
    sound = document.createElement("embed");
    sound.setAttribute("src", 'audio/' + fName + ".wav");
    sound.setAttribute("hidden", true);
    sound.setAttribute("autostart", true);
    document.body.appendChild(sound);
}
```

The shop-owners can stop playing the audio file by clicking on the stop icon. The following piece of JavaScript is used to stop playing the audio file:

```

function soundStop() {
    if (sound) {
        document.body.removeChild(sound);
        sound = null;
    }
}

```

The following section discusses the implementation of the navigation support system used to assist the shop-owners with how to navigate through the shop-owners' end.

4.10 Administrators' Help

The administrative help interface is a navigation support tool that makes use of a JavaScript to implement the collapsible and expandable functionality to assist the shop-owner with navigating easily through the shop-owners' end. An example of the administrative assistance interface is displayed in Figure 3.11 in the previous chapter. As mentioned in Chapter Three, this interface makes use of the system functionality diagram along with the system screenshots. However, the interface does not only make use of screenshots but also real-time *div* windows that the shop-owner can use to perform a certain functionality. The JavaScript used for the collapsible and expandable functionality also makes use of the JQuery treeview plugin. The JavaScript is as follows:

```

29     <script type="text/javascript" language="javascript">
30         $(function() {
31             $("#tree").treeview({
32                 collapsed: true,
33                 animated: "medium",
34                 control: "#sidetreecontrol",
35             });
36         });
37     </script>

```

From the JavaScript above, collapsed is set to true meaning the entire list of links will be collapsed by default. If collapsed was set to false, the list of all the links will be expanded by default. However, setting collapsed to true may cause misunderstanding with the targeted users. The drop down animation is set to medium but it can also be set to slow or fast based on

the speed at which the shop-owners prefer the animation to be played. This interface was implemented to assist the user in using the shop-owners' end of the shopping e-mall system.

4.11 Conclusion

This chapter has presented the implementation of the interventions which were mentioned in Chapter Three. These interventions include functionalities such as color adaptation, preferable language selection, audio clues, autocompletion and/or autosuggestion, and the navigation support tool. The following chapter discusses the findings and results obtained after observation and evaluation of the shop-owners' end. The findings and results were obtained after conducting interviews, questionnaires and a field test for the shop-owners' end.

CHAPTER FIVE

Analysis and Findings

5.1 Introduction

This research included constant exploration of the research field site, Dwesa, and required the intended users of the shop-owners' end of the shopping e-mall system to be interviewed and fill in questionnaires. This chapter discusses the analysis and findings based on the explorations, interviews and questionnaires. The aim of conducting interviews and using questionnaires with the shop-owners was, firstly, to collect data about the shop-owners perspective on which cultural markers can be implemented on the shop-owner interfaces. If there is any difference with the implementation needs between the interface designer and the shop-owners, changes would be made on the interfaces to reflect the cultural needs of the shop-owners. Secondly, the interviews and questionnaires were meant to assess the shop-owners' understanding and experience of using computers and allow them to self-evaluate the usability of the new shop-owner interfaces which we discussed earlier in this research.

5.2 System Evaluation

The following subsections discuss the assessment and findings of the shop-owner interfaces during field testing with the Ilingelihle Art and Craft group, based on the designer's perspective and on the participant's perspective. These findings were determined by evaluating the usability of the interventions implemented on the shop-owner interfaces and the look and feel of the interfaces.

5.2.1 Software Designers' Perspective

The design decisions include factors in the design such as user tasks, user types, layout structure, and data structure, thus improving the adaptability, flexibility, sustainability, and user experience. Because the shop-owner interfaces are a FOSS based e-Commerce software, the functionality and offered services can be extended as the software source code is freely available. Hence, the need to initially discuss the assessment, decisions taken, and findings on the ease and flexibility of implementing the software and its interventions to another deep rural setting from the developer's point of view.

5.2.1.1 Interventions

❖ **Language Selection:** as we realized in Chapter Three, using computers and the internet in the users' home language improves the users' understanding and experience. The language selection was implemented in a disciplined translation manner which could easily be translated in other regions. For example, if the system is deployed in another region where the population speaks Afrikaans, an Afrikaans translator would have to translate from English to Afrikaans. Hence, as we had mentioned in Chapter Three, every language has its own PO file. This means that there would be an Afrikaans PO file which is created in 3 easy steps:

1. Copy the original PO file stored in the English locale directory e.g. en_ZA (first two initials of the language + underscore + country domain for the language).
2. Paste the PO file in a new Afrikaans locale directory named, for example, af_ZA (Afrikaans).
3. Translate all the English words or sentences in the Afrikaans PO file to Afrikaans.

This demonstrates the ease and flexibility of translating the shop-owner interfaces from one language to another, not only in South African languages but also international languages.

❖ **Adaptation:** this refers to the cultural adaption of colours, icons, layout, themes, and shapes.

- ✓ **Colour** – colour adaptation in the system was implemented in a structured manner allowing the users to select a colour from a wide variety of colours. On this software, the colour picker is used to change the background colour of the interfaces. Implementing this functionality for another setting does not require any changes to be made on source code.
- ✓ **Icons** – the selection of the icons to be used is dependent on which language is currently used by the user. Icons differ among culture, tradition and interpretation or understanding. For example, icons that can be used for Xhosa culture may differ from that of Ndebele, Sotho, Venda, and other African cultures. This means:

1. A human icon creator/translator would be required to translate the icons to culturally localized equivalent icons of the targeted area where the software would be replicated.
 2. Assuming that the PO file for the language of the targeted area has been created following the 3 steps mentioned earlier, the localized icons are stored in a directory named after the locale of the selected language. For example, if the language is Swahili (i.e. locale = sw_KE), the directory for storing the localized icons for this culture should be named sw_KE.
 3. After this directory is created and the icon translation is complete, equivalent icons are given the same name. This is because the icon names do not change in the source code, but only the directories to which the icons are stored change.
- ✓ **Layout and Shapes** – the layout of the interfaces is able to be re-arranged to the layout preferred by the user. This is done by sorting it using a drag and drop functionality making it easy to configure to adapt in a different rural area with similarities to Dwesa. There are also several different layout shapes that the user can choose from.
 - ✓ **Themes** – Adding a new theme into the system means that a new CSS is introduced into the system. Introducing a new theme to the system is a bit complex because of the sortable layouts functionality. This is performed similarly to how the shop-owners upload a new item into their shop but in this case the CSS file is stored in the theme directory. If the uploaded CSS is designed using the same *id* and *class* names as the CSS currently used by the interfaces discussed in this research, the sortable layouts functionality will operate in the same way even for the new theme.
- ❖ **Edit-in-place:** edit-in-place interventions help decrease or limit the number of user interfaces and functionality separation. For example, usually when updating a database table or field, an extra interface is often required. This is not an organized form of updating the database because the user may intend to update a single cell of a table row but instead uses an interface that would allow updating of the whole row. Hence, this may not be the case for inexperienced computer users (such as the shop-owners) as navigation through the system interfaces is always problematic. The edit-in-place functionality was essential in this

research in cases such as user profile editing, item details editing, and also during the uploaded item preview. With edit-in-place, the user can easily update a table cell from a single interface without requiring the page to refresh.

- ❖ **Autosuggestion:** as some rural communities have members who are computer illiterate, they lack knowledge regarding the use of computer equipment (i.e. input devices such as keyboard, mouse, microphone, scanner, etc.) for communication and as a marketing resource. Due to the limited or non-existent computer experience and related skills encountered in marginalized rural communities, the possibility exists that community members may find it difficult to use these input devices. Therefore, for community members to be introduced to the use of computers without the need for training sessions there should be a user assistance tool from the system. Thus, the solution is to use autocompletion or autosuggestion to assist the users. Autosuggestion is slightly different from autocompletion because, instead of completing the word, it provides possible suggestions for the word typed and it is usually in a drop-down form. As the use of computer and technologies increases every day, new technologies are being invented and one would refer to autocompletion and autosuggestion as old hat even though it is still as effective for user assistance (Dyakalashe, 2008).
- ❖ **Audio Clues:** we experience audio, sound, voice communication in our everyday lives and we also commonly experience that people understand each other better when communicating in their home language. In our experience, elders and/or non-English speaking people in rural areas prefer listening to the radio broadcasts for news, radio programs, weather reports, and sports broadcasted in their home language instead of buying newspapers or magazines. Hence, it is important to develop interfaces with audio supportive tools for instructions, navigation, and notifications. The audio was translated in the two supported languages, English and isiXhosa. Similar to the language selection, each English audio file is stored in a separate directory to the isiXhosa audio files. Hence, if the user speaks English then the English audio files will be loaded during that session.

5.2.1.2 Layout Structuring

The structuring of the data and layout of the interfaces is done through the use of the drag and drop sortables functionality. This functionality allows users to sort the layout of the interfaces in a structure easily understood by them. Hence, adapting these interfaces in another culture is flexible and thus improving the adaptability and sustainability of the software in other regions.

5.2.2 Participants' Perspective

The following section discusses the analysis and findings of the tasks performed by the shop-owners during a field test evaluation. The analysis was conducted in the form of a training session with the shop-owners and involved a demonstration on how to use the shop-owner interfaces. This section is divided into two subsections and discusses the tasks which the shop-owners performed whilst also mentioning some challenges faced by the shop-owners during the exercise. The aim of the evaluation exercise was to determine whether the implemented cultural markers and intervention improved the usability of the system and that the users understand the shop-owner interface usage.

Firstly, the training and system demonstration sessions were conducted with the Ngwane Art and Craft shop-owners residing at Dwesa and after the demonstration session the shop-owners were invited to use the system under the trainers' supervision. Because Dwesa is a deeply rural environment, organizing interviews or meetings with the community members is still a difficult process as most of the community members (i.e. in this case the shop-owners) are located far away from the two schools. Hence, there were a relatively small number (30) of questionnaire forms which were distributed and fully completed. However, due to computer and network failures, we were forced to conduct the sessions with a laptop and projecting the screen to the wall with an overhead projector. In light of this, a less formative approach was used for collecting their points of view about the shop-owner interfaces.

An equivalent training and system demonstration session was also conducted with the other registered shop-owner, the Ilingelihle Art and Craft group in Alice. The sessions were conducted in one of the labs at the University Of Fort Hare Department Of Computer Science. There were

approximately five members of the Ilingelihle group that managed to participate in the session. The training and system demonstration sessions took approximately three hours and were conducted in the shop-owners' home language, isiXhosa. After the sessions were completed, the shop-owners suggested that they would prefer to have another demonstration session as the three hours was too long for them to concentrate and learn about so many things. Because of the poor ability of concentration and remembering, a short exercise similar to the main exercise they were going to work on later that day was conducted. Some of the results discovered regarding the use of shop-owner interfaces from the shop-owners, after conducting the questionnaires, include the following:

- ❖ As Dwesa is a deeply rural environment in the Eastern Cape, the participants (students, teachers, and shop-owners) of the questionnaire were all using isiXhosa as their official home language. However, the students and school teachers preferred the use of either isiXhosa or English for the interface language communication. The participants use isiXhosa to communicate with the other community members and sometimes they would also use English to communicate between students and school teachers.
- ❖ Approximately 67% of the participants of the questionnaire strongly agreed that the use of the home and native language would make it more interesting to use computers and the internet. Hence, 40% of them strongly agreed that in the future, more multilingual websites should be designed to accommodate multilingual societies in different countries (i.e. South Africa and other countries in Africa and overseas) whilst 30% only agreed and 17% were not sure whether this should be implemented.
- ❖ The participants were asked about their preferences of background colours (i.e. from this list – black, blue, green, grey, yellow, orange, purple, red, and white or specify another colour) and they had an option of choosing more than one colour for this question. The answer chart with participants' ratings for this question was as follows:

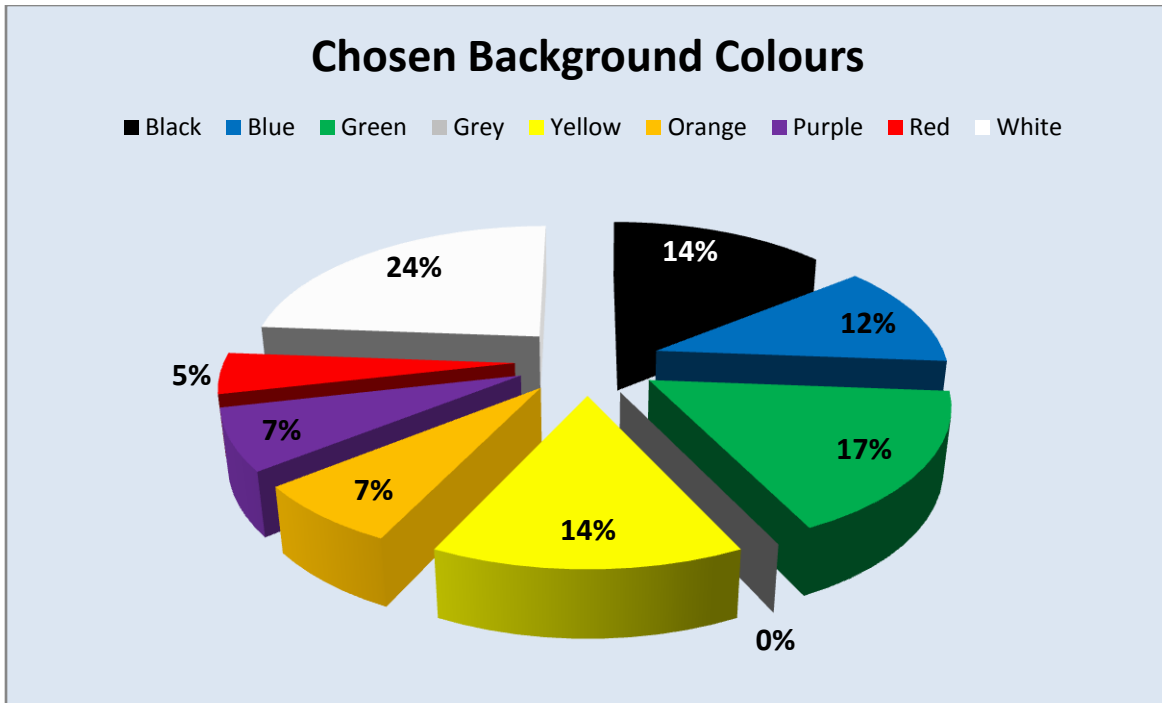


Figure 5.1: Statistics of Background Color Preferences by Questionnaire Participants

- ❖ After explaining what is meant by font colour, the participants were also asked about their preferable font colours. The participants' font colour choice chart and statistics were as follows:

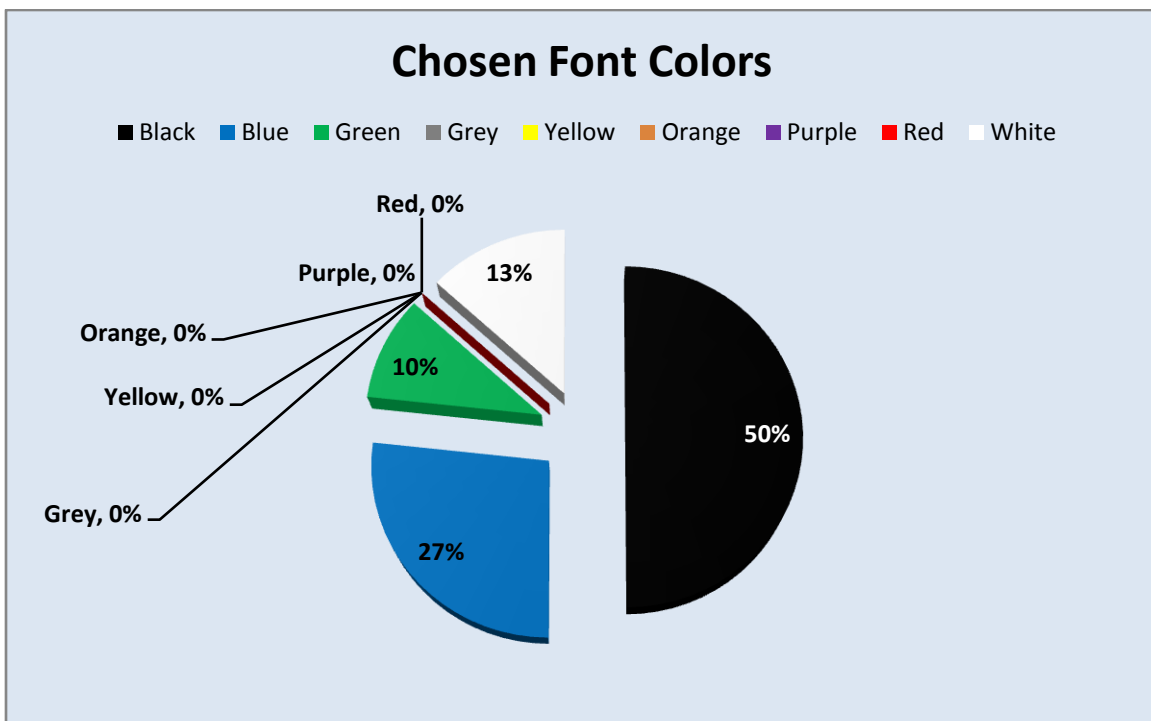


Figure 5.2: Statistics of Font Color Preferences by Questionnaire Participants

45% of the participants strongly agreed that the use of graphic and primarily less textual interfaces (i.e. those which primarily make use of audio, text, pictures, icons, symbols, metaphor, etc.) are important for the usability of the interfaces. 33.3% of the participants were not sure about what is actually meant by terms like graphical and textual interfaces.

The results from the following sections are based on the findings we obtained after the exercises with the shop-owners from the Ilingelihle Art and Craft groups.

5.2.2.1 Browsing Uploaded Items

After the users logged in as shop-owners, their first exercise was to browse and search through the image slideshow and view the uploaded items. The shop-owners were able to browse through the slideshow and also understood that to view an items' details (i.e. item name, price, weight, description and a zoomed in item image) on the image gallery, they would have to click on the image from the slideshow. They felt it was easier for them to use a mouse flying over effect than when they would have to click to trigger the slideshow. During this exercise, both groups understood the tasks they had to perform. The Ilingelihle Art and Craft shop-owners managed to execute the tasks at hand. From this exercise, it was established that there is a high probability that the shop-owners can manage to browse and search through their uploaded items from the slideshow.

5.2.2.2 Edit-in-Place

After they clicked an item thumbnail on the slideshow, they saw an enlarged image of that thumbnail with the details of that item on the image gallery. For this exercise the shop-owners were requested to change/modify the name, image, price, description, and mass of the item they had selected. The shop-owners managed to edit the item name, price, description, and mass but had problems understanding how to change the item's image. It was determined that the reason for this was the fact that they could not navigate through to the destination folder or file. Another problem was that they could not preview the destination file before actually selecting it. Hence, it was compulsory that in future the procedure for uploading item images should be in the form of an image uploading application that allows users to see a preview (i.e.

as a list of thumbnails) of all the images inside a certain directory they have chosen. An example of such an application would be the java-based photo uploader tool used by Facebook to upload photos (Facebook, 2009).

5.2.2.3 Selecting a Language

As we mentioned earlier in the research, the localized shop-owner interfaces are currently operating in two South African languages, i.e. English and isiXhosa. During this exercise, it was observed that the users preferably used isiXhosa. The Ilingelihle group informed us that they had a computer training session with some American visitors who were interested in their artifacts. However, they could not follow or understand what they were being taught at the training sessions because of the advanced language the American trainers were using during the sessions. However, our training and demonstration sessions were conducted in isiXhosa so that the trainers' instructions are easily understood as isiXhosa is their home language. Hence, during this exercise, the shop-owners were able to recall the language selection instructions they were taught during the demonstration thus making it simpler for them to select their preferable language which is isiXhosa.

5.2.2.4 Deleting and Uploading Items

Understanding how this section of the software operates is very important for the shop-owners as it is the only way they can load items into their shops. The shop-owners did not experience many problems in understanding how the uploading form is filled in. The upload form has automatic item name suggestion which they felt was necessary as they often type the names incorrectly. However, as it was mentioned earlier, the only problem they encountered was when they had to upload the item images hence it is important to find a tool that would simplify the uploading of images for the correct items.

5.2.2.5 Colour Selection and Sortables

The shop-owners were excited with using the colour picker for selecting their preferable colours and it held their attention. The reason for this is because the shop-owners often work with colourful beads for beading their artifacts. They quickly understood how to use the colour

picker. At first they had a problem with dragging the two colour markers to the colours they intended to blend although it became easier after some time. This was because the shop-owners were not familiar with using the mouse and still required more computer training sessions. Because the colour selection was a trial and error exercise, it was observed that the shop-owners are very likely to perform this task in future without requiring any aid.

The rigidity of dragging and dropping by the shop-owners also caused a few problems when they were sorting the layout of the interfaces. But because the shop-owners lack computer literacy and have not been using the internet often, they were not too concerned with the re-arrangement of the interface layout. Therefore, because the system is an open source software that could be replicated for other cultures, this functionality would require extra options for locking or unlocking the sortable layout of the interfaces.

5.2.2.6 Viewing Orders

When an order is made in the shop, it triggers an email notification to the shop-owners who can also view the orders from their online stores. During the sessions, the shop-owners were shown how an order is made by the customers and how the order is displayed from their interfaces. The shop-owners were able to view the orders and the contact details of the customer who had made the order for updates on the delivery of an order. They understood that after successfully delivering an order, they would have to change the order status from the system so that they do not mistakenly deliver the same order to the same customer twice or more times.

5.3 Discussion

One reason for implementing an e-Commerce shopping mall system in rural areas, such as Dwesa, is that some rural communities have groups of individuals who have the skill to produce hand-crafted artifacts but lack the skills and resources for advertising and marketing them. Based on the interviews with the shop-owners and observations of the shopping e-mall system, it was noticed that the first version (non-localized) of the shop-owner interfaces had great cultural and linguistic limitations.

The aim of the project was to localize and improve the usability of the user interface of the e-Commerce system on the shop-owners' end, both linguistically (in isiXhosa, a language used by the shop-owners at Dwesa) and culturally (graphic representation/locally developed arts and crafts artifacts). The final system design was based on the feedback and input from the interviews and questionnaires conducted with the shop-owners. This was done by studying the cultural needs of the prospective users of the platform. We investigated the specific localization elements or interface design elements that are preferable for a cultural community such as Dwesa. These design elements are widely referred to as cultural markers. We have discussed the use of cultural markers for improving the accessibility and usability of ICT services targeted for rural development.

5.4 Conclusion

The results attained from the field test highlights that localization of software designed for rural marginalized communities is a very important process of improving the user-attractiveness, user-friendliness, and usability of web interfaces. In this research, we discussed the observations of the design and re-implementation of user-attractive, user-friendly, meaningful shop-owner interfaces of the deployed e-Commerce system for the local entrepreneurs at Dwesa. The shop-owner interfaces are designed so that when the system is replicated in another region, the users can easily configure the interfaces according to their culture and language. The research also focused on the software adaptability, sustainability, and usability issues encountered in rural environments because of low-level or limited rates of literacy and considering the fact that the users are old people with little to no computer literacy. It is noticeable that culturally and linguistically localizing the interfaces allows the shop-owners to easily manage their stores.

CHAPTER SIX

Conclusion

6.1 Research Summary

The aim of this research was to localize and improve the usability of the user interfaces of the e-Commerce system designed on the shop-owners' end, both linguistically (i.e. in isiXhosa, a language people in rural areas in the Eastern Cape province use) and culturally (graphic representation of the shop-owner interfaces). The shop-owner interfaces are currently operating in two official languages, English and isiXhosa but in the future they will support all eleven South African languages and other African languages. For this research, the shop-owner interfaces were initially designed in reflection of the authors' cultural and linguistic understanding of the customs and history of the Xhosa culture. However, it was important that the author learned and understood the cultural and linguistic needs of the targeted users. Therefore, after obtaining the feedback from the questionnaires and interviews, the shop-owner interfaces were designed with the users' cultural and linguistic needs and interest in mind.

This was also done by studying the needs of the prospective shop-owners using the platform. To design the shop-owner interfaces, we had to determine specific localization elements or interface design elements that are preferable for a cultural community such as Dwesa. These design elements are normally referred to as cultural markers. We mentioned in the previous chapter that the cultural markers increase and improve the usability of a system. The merging of culture and usability in a web design directly impacts the users' perception of the credibility and trustworthiness of the website. The merging of culture and usability is known as *Culturability*.

The focus of the research was on the design and re-implementation of the shop-owner interfaces considering the ease of adaptability, sustainability and usability of the interfaces for different environments. The design and display of information for a specific area or environment may differ in effectiveness and usability because of the difference in culture, not just for the buyer but for the sellers as well. In the case of the shopping e-mall system deployed in Dwesa, there is a vast gulf between the buyers and sellers. The shopping e-mall system was

developed as a FOSS because the services of this software would also be simulated in other regions of South Africa and in Africa with a similar background as Dwesa.

The system currently has two online stores (i.e. two art and craft groups) and will also include many more other shops in the future. One of the shops is called the *Ngwane Art and Craft shop* and its group members are situated at Dwesa. The group specializes in producing beaded art and craft ranging from beaded anklets and necklaces to traditional clothes with colourful African beads. The other shop is called the *Ilingelihle Art and Craft shop* and is located at the University of Fort Hare in Alice, Eastern Cape. They produce beaded arts and crafts as well as pillow cases (including continental), traditional bed sheets, table cloths, traditional bags (e.g. toiletry bag and book bags), and many more artifacts. These artifacts can be browsed through and bought from the “buy at Dwesa” website (<http://www.dwesa.com>).

Some of the artifacts designed by the *Ilingelihle Art and Craft group* were ordered at the 2008 SAICSIT conference held at the Wilderness Beach Hotel and Spa, Garden Route, Wilderness, South Africa. The conference organizers bought 130 traditionally beaded pencil cases for the people whose papers were accepted at the conference. A picture of one of the pencil cases is displayed in figure below, Figure 6.1.



Figure 6.1: Pencil Cases sold to the SAICSIT 2008 Conference

We have had a few problems with advertising the website, but some advertising ideas were shared amongst the Dwesa research team. Attending in conferences (i.e. Southern African Telecommunication Networks and Applications Conference – SATNAC 2008 and the 2008 THRIIP award ceremony) has also helped with advertising the website. The artifacts were actually ordered after presenting the software at SATNAC 2008 and attracted the SAICSIT 2008 conference organizers. However, there is still some work that needs to be done in terms of marketing the website.

6.1.1 Summary by Objectives

- ❖ **ICT and L10N:** we have investigated and proved (through the use of interviews and questionnaires) that localizing deployed ICT services better improves the user's understanding of how to use these services which can economically improve their community and also how they are able to benefit from them. Lack of education background together with minimal information and communication resources are the influential factor for rural community marginalization. Localization breaks these barriers by considering the translation and localizing of the ICT services to the user's culture and language, both textual and verbal.
- ❖ **Cultural Diversity:** we have discussed that cultural differences also have an impact on the ease of understanding the usefulness of ICTs for rural development. For example, the use of a specific picture and/or icon may not symbolize or have the same meaning. Thus it is important to study the cultural needs of the culture or community the developer is targeting.
- ❖ **Usability:** we have investigated and proven that through the use of interviews, questionnaires, and field tests that the shop-owners' experience and understanding of using the system has improved due to localized interfaces.
- ❖ **Self-reliance:** the results discussed in Chapter Five proved that the use of cultural markers in web design has improved the shop-owners' understanding of how to monitor and sustain their stores by themselves.

6.2 Future Work

As mentioned earlier, the software discussed in this research is a FOSS based e-Commerce prototype. This means that there are many possible extensions that could be made as the source code is freely available. Here are some extensions:

- ❖ Finding a working payment system that will accept credit card payment in the South African currency (Rands) thus eliminating the need for currency conversion.
- ❖ The implementation of audio or voice I/O interactive interfaces with the shop-owners in the form of text-speech communication and automatic speech recognition.
- ❖ Supporting multiple messaging or notification services such as SMS after an order is made.
- ❖ Creating video tutorials that could be used to assist first time users of the shop-owner interfaces in using the interfaces. If the user is not new to the system, they can skip the tutorial and continue with their usual tasks.
- ❖ Finding a free and open source application or tool that will simplify the uploading of item images during the process of uploading new items into shops. An example of such an application or tool, given in Chapter Five, would be a similar application to the Facebook Photo Uploader.
- ❖ Audio and video clip uploading for supporting micro-tourism (e.g. how the advertised items are designed, travel tour guide, explanation of the traditional meaning of the items, etc.).

6.3 Conclusion

In this research we discussed the motivation and need for designing and re-implementing culturally and linguistically localized user-attractive, user-friendly, and meaningful shop-owner interfaces of their virtual online stores for rural development. The discussed software is designed to flexibly adapt to other rural marginalized settings upon replication. The shop-owners can easily configure the interfaces using the interventions discussed in the research so as to adapt according to their culture and language. Through the use of interviews, questionnaires, field visits, and field testing, we observed that culturally and linguistically localizing the interfaces improves the shop-owner's experience and understanding. Localization

simplifies the usability of the interfaces, allowing the shop-owners to easily manage their registered stores.

This research not only considered localization but also covered some level of internationalizing and globalizing the discussed software. Localization, internationalization, and globalization have very strong similarities and because the software can be deployed in other African countries, we applied a combination of the processes in the re-development of the shop-owner interfaces. From the implementation chapter, we observed that the objectives of the research have actually been carried out successfully. The design and implementation chapters demonstrated and discussed the changes made to the old version of the shop-owner interfaces.

The people involved in both art and craft groups were given basic ICT literacy skills to a level allowing them to self-monitor and manage their online stores. The group members were trained and demonstrations were given on how to use the system enabling them to upload new items, check customer orders, edit item details, etc. After the training sessions, demonstrations and explanations about how the e-Commerce platform operates, both groups had an understanding of the marketing benefits and skills to be gained from using such a system. The training sessions were conducted in isiXhosa and this made the sessions a lot easier to understand.

Bibliography

Ambra, R., & Ruta, F. (2009). The Evolutionary Software Development Process used in the Upgraded AMX Human Machine Interface Design. Retrieved December 10, 2009, from NATO Research and Technology Organization: <http://ftp.rta.nato.int/public//PubFullText/RTO/MP/RTO-MP-102///MP-102-12.pdf>

Anton, A. I., Carter, R. A., Earp, J. B., & Williams, L. A. (2001). EPRAM: Evolutionary Prototyping Risk Analysis & Mitigation (e-Commerce Software Development Process Document). Technical Report #TR-2001-08, North Carolina State University, Department of Computer Science.

Barber, W., & Badre, A. (1998). Culturability: The Merging of Culture and Usability. Proceedings of the 4th Conference on Human Factors and the Web - Our Global Community. Basking Ridge, New Jersey, USA.

Beshar, B. (2008). Machine Translation vs Translation Memory Tools. Retrieved November 15, 2008, from Illinois I.T. Association, Inc: <http://www.illinoistech.org/technologynews.aspx>

Briggs, B. (2007). An Introduction to Translation and Localization for the Business Executive. Retrieved January 13, 2009, from Acclaro Inc: http://www.acclaro.com/assets/files/downloads/whitepapers/intro_localization_for_busy_executive.pdf, pp. 3-7

Carr, M., & Verner, J. (1997). Prototyping and Software Development Approaches. Retrieved December 10, 2009, from http://rguerrero334.blogspot.es/img/Prototyping_and_Software_Development_Approaches.pdf

Coghlan, D., & Brannick, T. (2005). Doing Action Research in Your Own Organization (2nd Edition ed.). London: SAGE Publications Ltd.

Craciunescu, O., Gerding-Salas, C., & Stringer-O'Keeffe, S. (2004). Machine Translation and Computer-Assisted Translation: A New Way of Translating? Retrieved January 12, 2009, from Translation Journal, Vol. 8, No. 3: <http://www.accurapid.com/journal/29computers.htm>

Cyr, D., Carole, B., & Joe, I. (2004). Design and E-loyalty Accross Cultures In Electronic Commerce. Proceedings of the 6th International Conference on Electronic Commerce, (pp. 351-360). Delft, Netherlands, 25-27 October.

Dalvit, L., Terzoli, A., & Wolff, F. (2008). Open-Source Software and Localization in Indigenous South African Languages with Pootle. Southern African Telecommunication Networks and Applications Conference - SATNAC. Wild Coast Sun, Eastern Cape, South Africa, 7-10 September.

Dalvit, L., Terzoli, A., Muyingi, H., & Thinyane, M. (2007). The Deployment of an E-commerce Platform and Related Projects in a Rural Area in South Africa. *International Journal of Computing and ICT Research*, Vol. 1 (No. 1), pp. 9-17, 13-15 June.

Dohler, P. N. (1997). Facets of Software Localization - A Translator's View. Retrieved March 31, 2009, from *Translation Journal*, Vol. 1, No. 1: <http://www accurapid.com/journal/softloc.htm>

Drepper, U., Meyering, J., Pinard, F., & Haible, B. (2007). Native Language Support Library and Tools. Retrieved July 10, 2008, from GNU Gettext Tools: <http://www.gnu.org/software/gettext/manual/gettext.pdf>

Duggan, B., & Deegan, M. (2003). Considerations in the Usage of Text To Speech (TSS) in the Creation of Natural Sounding Voice Enabled System. *ISICT: Proceedings of the 1st International Symposium on Information and Communication Technologies*, Vol. 49, pp. 433-438. Dublin, Ireland, 24-26 September.

Dyakalashe, S., Muyingi, H. N., Terzoli, A., & Thinyane, M. (2008). Cultural and Linguistic Localization of the Shop-Owner Interfaces to an E-commerce Platform for Rural Development. *Southern Africa Telecommunication Networks and Applications Conference - SATNAC*. Wild Coast Sun, Eastern Cape, South Africa, 7-10 September.

Esselink, B. (1998). *A Practical Guide to Localization* (1st Edition ed.). Amsterdam, Netherlands: John Benjamin Publishing Company.

Facebook. (2009). Simple Photo Uploader. Retrieved November 3, 2009, from Facebook: <http://www.facebook.com/apps/application.php?id=2371229427>

Ferreira, R. (2002). Culture and E-commerce: Culture Based Preference for Interface Information Design. M.Sc Dissertation, Virginia Polytechnic Institute and State University, Industrial and Systems Engineering, Blacksburg, Virginia, USA.

Ford, G. (2005). Researching the Effects of Culture On Usability. M.Sc Dissertation, University of South Africa, Department of Information Systems.

Forkjaer, E., Hertzum, M., & Hornbaek, K. (2000). Measuring Usability: Are Effectiveness, Efficiency, and Satisfaction Really Correlated. *Proceedings of the Computer Human Interaction - CHI 2000 on Human Factors In Computing Systems*. The Hague, The Netherlands, 1-6 April.

Frimannsson, A. (2005). Adopting Standards Based XML File Formats In Open Source Localization. B.IT (Honours) Thesis, Queensland University of Technology, School of Software Engineering and Data Communications, Brisbane, Queensland, Australia.

Giannini, D. (2004). ICT and Poverty Reduction in the Sub-Saharan Region. *Adult Education and Poverty Reduction Conference*, (pp. 151-157). University of Botswana, Botswana, 14-16 June.

Goetze, M., & Strothotte, T. (2001). An Approach to Help Functionally Illiterate People with Graphical Reading Aids. Proceedings of the 1st Symposium on Smart Graphics, (pp. 39-42). New York, USA, 21-23 March.

Golding, P., Donaldson, O., Tennant, V., & Black, K. (2008). An Analysis of Factors Affecting the Adoption of ICT by MSMEs in Rural and Urban Jamaica. 16th European Conference on Information Systems - ECIS, (pp. 1286-1297). Galway, Ireland, 9-11 June.

Gordon, S. V., & Bieman, J. M. (1993). Reported Effects of Rapid Prototyping on Industrial Software Quality. Software Quality Journal , Vol. 2 (No. 2), pp. 93-110.

Green, D., & DiCaterino, A. (1998). A Survey of System Development Process Models. Retrieved December 8, 2009, from Centre for Technology in Government: http://www.ctg.albany.edu/publications/reports/survey_of_sysdev/survey_of_sysdev.pdf

Harris, R. W. (2004). Information and Communication Technology for Poverty Alleviation. pp. 8-10. Kuala Lumpur, Malaysia: United Nations Development Programme's Asia-Pacific Development Information Programme (UNDP-APDIP).

Heerden, v. C., Barnard, E., & Davel, M. (2009). Basic Speech Recognition for Spoken Dialogues. 10th Annual Conference of the International Speech Communication Association (Interspeech 2009), (pp. 3003-3006). Brighton, UK, 6-10 September.

Hofstede, G. (1997). Cultures and Organizations: Software of the Mind, Intercultural Cooperation and Its Importance for Survival. McGraw Hill, New York, USA.

iGoogle. (2009). iGoogle. Retrieved November 30, 2009, from Google: <http://www.google.com/ig>

International Organization for Standardization - ISO 9241. (1998). ISO 9241-11: Guidance to Usability.

Jakachira, B. T. (2008). (University of Fort Hare) Retrieved November 20, 2008, from Dwesa Government Online Service: <http://www.dwesa.com/egov/>

Jakachira, B. T., Muyingi, H. N., & Wertlen, R. (2008). Implementation of a Web-Based E-government Proxy for a Marginalized Rural Population. Southern Africa Telecommunication Networks and Applications Conference - SATNAC. Wild Coast Sun, Eastern Cape, South Africa, 7-10 September.

Keniston, K. (1997). Software Localization: Notes on Technology and Culture. Working Paper #26, Massachusetts Institute of Technology, Program in Science, Technology, and Society, Cambridge, Massachusetts, USA.

Khaslavsky, J. (1998). Integrating Culture into Interface Design. Proceedings of the Computer Human Interaction - CHI 1998 on Human Factors In Computing Systems. Los Angeles, California, USA, 18-23 April.

Kirakowski, J., & Corbett, M. (1993). SUMI: The Software Usability Measurement Inventory. *British Journal of Educational Technology* , Vol. 24 (No. 3), pp. 210-212.

LAMP- Linux, Apache, MySQL, and PHP. (2009). Retrieved February 2, 2009, from Wikipedia: [http://en.wikipedia.org/wiki/LAMP_\(software_bundle\)](http://en.wikipedia.org/wiki/LAMP_(software_bundle))

Launchpad Translation. (2008). (Canonical Launchpad Engineering) Retrieved November 16, 2008, from Launchpad: <https://launchpad.net/rosetta>

MacManus, R. (2007). Google's Personalization Push: iGoogle, Localization, Gadget Maker. Retrieved November 30, 2009, from Read Write Web: http://www.readwriteweb.com/archives/google_personalization_push_igoogle_search.php

Majewski, M., & Kacalak, W. (2005). Intelligent Human-Machine Speech Communication System. *International Conference on Intelligent Computing - ICIC 2005*, (pp. 3441-3450). Hefei Anhui, China, 23-26 August.

Mandioma, M. T. (2007). Rural Internet Connectivity: A Deployment in Dwesa-Cwebe, Eastern Cape, South Africa. M.Sc Dissertation, University of Fort Hare, Department of Computer Science, Alice, Eastern Cape, South Africa.

Masoeu, A., & de Villiers, C. (2001). Web Usability in a Multicultural Environment: A Concern for Young South African Web Users? *Conference on Human Computer Interaction in Southern Africa - CHI-SA*. Pretoria, South Africa, 10-12 September.

Mehta, S., & Kalra, M. (2006). Information and Communication Technology: A Bridge for Social Equity and Sustainable Development. *The International Information and Library Review* , Vol. 38 (No. 3), pp. 147-160.

Najjar, L. J. (1990). Rapid Prototyping (TR 52.0020). Atlanta, Georgia, USA: IBM: Corporation.

Ngcobo, P., & Herselman, M. E. (2007). Evaluating ICT Provision in Selected Communities in South Africa. *Information and Beyond: The Journal of Issues in Informing Science and Information Technology* , Vol. 4 (No. 1), pp. 713-724.

Nielsen, J. (1993). *Usability Engineering*. Academic Press.

Njeje, S. G. (2007). Dwesa E-commerce Shopping Mall. (University of Fort Hare) Retrieved November 12, 2008, from buy at Dwesa: <http://www.dwesa.com>

Njeje, S. G. (2007). Implementing a Robust, Cost Effective, E-commerce Platform for a Disadvantaged Community of the Eastern Cape, South Africa. M.Sc Dissertation, University of Fort Hare, Department of Computer Science, Alice, Eastern Cape, South Africa.

Ó Slatara, A. (2002). An Introduction to the Localization of eContent. *Localization Research Centre - LRC 7th Annual Localization Conference On eContent Localization*. Dublin, Ireland, 12-13 November.

Organisation for Economic Co-Operation and Development - OECD. (2005). Good Practice Paper on ICTs for Economic Growth and Poverty Reduction. The Development Assistance Committee - DAC Journal 2005 , Vol. 6 (No. 3).

Organisation for Economic Co-Operation and Development - OECD. (2003). ICT and Economic Growth: Evidence From OECD Countries, Industries and Firms. Paris, France: OECD Publications Service.

Osborn, D. (2007). A Survey of Localisation in African Languages, and its Prospects: A Background Document. Retrieved November 30, 2009, from PAL - Pan African Localization: http://www.panafril10n.org/wikidoc/uploads/PAL_Document_Oct_2007.pdf

Osborn, D. (2005). PAN Africa Localization: Capacity Building for Production of Software and Content in Arabic and African Languages. Retrieved November 27, 2009, from International Development Research Centre - IDRC: http://www.idrc.ca/en/ev-87747-201-1-DO_TOPIC.html

Osborn, D. Z. (2004). African Languages and Information and Communication Technology (ICT): Key Elements for the Future. Fourth Conference on Preserving African Languages, (pp. 96-110). Eastern Shore, Salisbury, USA, 4-7 November.

Paypal. (2009). Retrieved November 19, 2009, from Paypal: <https://www.paypal.com/za>

PHP MySQL Functions. (2009). Retrieved February 2, 2009, from W3Schools: http://www.w3schools.com/PHP/php_ref_mysql.asp

Plauché, M., Nallasamy, U., Pal, J., Wooters, C., & Ramachandran, D. (2006). Speech Recognition for Illiterate Access to Information and Technology. International Conference on Information and Communication Technologies and Development. School of Information at the University of California, Berkeley, USA, 25-26 May.

Pluke, M., Peterson, F., Pollard, D., & Szalai, B. (2006). Cultural Inclusion in Information and Communication Services. 20th International Symposium on Human Factors in Telecommunication. Sophia-Antipolis, France, 20-23 March.

poEdit Homepage. (2005). Retrieved November 14, 2008, from poEdit: <http://www.poedit.net>

Pootle. (2008). (Translate.org.za) Retrieved November 8, 2008, from Wikipedia: <http://en.wikipedia.org/wiki/Pootle>

Pretorius, L., & Bosch, S. E. (2003). Enabling Computer Interaction in the Indigenous Languages of South Africa: The Central Role of Computational Morphology. Interactions: Winds of Change , Vol. 10 (No. 2), pp. 56-63.

Rajesh, M. (2003). A Study of the Problems Associated with ICT Adaptability in Developing Countries in the Context of Distance Education. Turkish Online Journal of Distance Education , Vol. 4 (No. 2).

Resig, John and the JQuery Team. (2008). JQuery Homepage. Retrieved November 09, 2009, from JQuery: <http://jquery.com>

Sasikumar, M., Aparna, A., Naveen, K., & Rajendra, M. P. (2005). Free/Open Source Software. Retrieved November 8, 2008, from Guide to Localization: <http://anakin.ncst.ernet.in/~aparna/consolidated/index.html>

Scott, M. S., Muyingi, H. N., & Thinyane, M. (2008). Investigation and Development of an E-judiciary Service for a Citizen-Oriented Judiciary System in Rural Community. Southern Africa Telecommunication Networks and Applications Conference - SATNAC. Wild Coast Sun, Eastern Cape, South Africa, 7-10 September.

Siyakhula Living Lab. (2008). Retrieved November 20, 2008, from <http://www.dwesa.org>

Software Prototyping. (2009). Retrieved December 8, 2009, from Wikipedia: http://en.wikipedia.org/wiki/Software_prototyping

Sommerville, I. (2000). Software Engineering. Retrieved December 8, 2009, from Lancaster University Computing Department: <http://www.comp.lancs.ac.uk/computing/resources/lanS/SE6/Slides/PDF/ch8.pdf>

South African Reserve Bank. (2008). Retrieved November 12, 2008, from <http://www.reservebank.co.za>

Sun, H. (2001). Building A Culturally-Competent Corporate Web Site: An Exploratory Study of Cultural Markers in Multilingual Web Design. Special Interest Group on Design of Communication - SIGDOC: Proceedings of the 19th Annual International Conference on Computer Documentation, (pp. 95-102). New York, USA.

Sun, H. (2004). Expanding the Scope of Localization: A Cultural Usability Perspective on Mobile Text Messaging Use in American and Chinese Contexts. PhD Dissertation , Rensselaer Polytechnic Institute, Department of Philosophy, Troy, New York, USA.

Tarwireyi, P., Terzoli, A., & Thinyane, M. (2007). Implementation of an Internet Access Cost Management System for Disadvantaged Communities. Southern Africa Telecommunication Networks and Applications Conference - SATNAC. Sugar Beach Resort, Mauritius, 9-13 September.

Translate Toolkit & Pootle. (2008). Retrieved November 16, 2008, from Translate Toolkit & Pootle: <http://translate.sourceforge.net/wiki/pootle/features>

Translation Memory. (2008). Retrieved November 15, 2008, from Wikipedia: http://en.wikipedia.org/wiki/Translation_Memory

Turker, R. C., Hickey, M., & Haddock, N. (2003). Speech Technology and Localization. Personal and Ubiquitous Computing , Vol. 7 (No. 1), pp. 22-29.

Vanhouwaert, B. (2002). Rosetta Manual. Retrieved November 11, 2009, from <http://www.irule.be/bvh/c++/rosetta/doc/rosetta.pdf>

Vatrapu, R., & Pérez-Quinones, M. A. (2006). Culture and Usability Evaluation: The Effects of Culture in Structured Interviews. *Journal Of Usability Studies*, Vol. 1 (No. 4), pp. 156-170.

Weir, G. R., & Lepouras, G. (2001). Localization and Linguistic Anomalies. *Proceedings of Human Computer Interaction - HCI International*, (pp. 535-539). New Orleans, USA.

Wittens, S. (2007, January 8). Farbtastic. Retrieved November 18, 2008, from Farbtastic: <http://acko.net/dev/farbtastic>

Appendices

Appendix A

SHOP OWNER INTERFACES CULTURAL MARKER REQUIREMENTS EVALUATION

This questionnaire is meant to be filled in by the local community members (i.e. teachers, students, and the targeted shop owners) at Dwesa. The aim of the questionnaire is to collect data pertaining to cultural markers which can be used on the shop owner interfaces for improving the interfaces' usability. The data collected will be used in the re-development of the shop owner interfaces to a culturally and linguistically localized e-commerce platform for rural development.

1. What is your age range from the ranges listed below?

- A. Less than 30 years
- B. 30 – 40 years
- C. 40 – 50 years
- D. 50 – 60 years
- E. More than 60 years

2. How many years have you been using computers and the internet?

- A. Never before
- B. Less than a year
- C. 1 - 2 years
- D. 2 – 3 years
- E. More than 3 years

3. What is your official native language?

- A. Afrikaans
- B. English
- C. isiNdebele
- D. isiXhosa

- E. Northern Sesotho
- F. Southern Sesotho
- G. Swati
- H. Tsonga
- I. Tswana
- J. Venda
- K. Zulu
- L. Other (unofficial languages):

4. Which of the official languages do you prefer to use? (You can choose more than one option).

- ☐ Afrikaans
- ☐ English
- ☐ isiNdebele
- ☐ isiXhosa
- ☐ Northern Sesotho
- ☐ Southern Sesotho
- ☐ Swati
- ☐ Tsonga
- ☐ Tswana
- ☐ Venda
- ☐ Zulu
- ☐ Other (unofficial languages):

5. The language(s) you have chosen in question 3 is/are the language(s) you use to communicate with other local community members?

- A. Yes
- B. No

6. Do you think that using computers and the internet in your native language and cultural needs would interest you more in using computers and the internet more often?
- A. Strongly Agree
 - B. Agree
 - C. Not Sure
 - D. Disagree
 - E. Strongly Disagree
7. Should computers and computer services be translated into different languages so that a user can use his/her language of choice when using computers?
- A. Strongly Agree
 - B. Agree
 - C. Not Sure
 - D. Disagree
 - E. Strongly Disagree
8. Which colour would you prefer to use for the background of the interfaces for the system?
- A. Black
 - B. Blue
 - C. Green
 - D. Grey
 - E. Yellow
 - F. Orange
 - G. Purple
 - H. Red
 - I. White
 - J. Other:
9. What does the colour you selected on question 8 represent or symbolize?
-
-

.....
.....

10. Which font colour would you prefer to use for the text used in interfaces of the system?

- A. Black
- B. Blue
- C. Green
- D. Grey
- E. Yellow
- F. Orange
- G. Purple
- H. Red
- I. White
- J. Other:

11. Do you think that allowing users to choose their preferable colours on the interface would improve the attractiveness, friendliness, meaning, and usability of the interfaces?

- A. Strongly Agree
- B. Agree
- C. Not Sure
- D. Disagree
- E. Strongly Disagree

12. Do you think that using pictures, icons, symbols, metaphors, etc instead of text can improve the ease of use, understanding, and communication between the user and the interfaces?

- A. Strongly Agree
- B. Agree
- C. Not Sure
- D. Disagree
- E. Strongly Disagree

13. The types of interfaces I prefer to use are those with many pictures and graphics instead of text?

- A. Strongly Agree
- B. Agree
- C. Not Sure
- D. Disagree
- E. Strongly Disagree

14. The types of interfaces I prefer to use are those that are primarily textual instead of having too many graphics?

- A. Strongly Agree
- B. Agree
- C. Not Sure
- D. Disagree
- E. Strongly Disagree

15. The types of interfaces I prefer to use are those that have audio/sound interactions with the user (e.g. text-to-speech and/or speech recognition)?

- A. Strongly Agree
- B. Agree
- C. Not Sure
- D. Disagree
- E. Strongly Disagree

16. Software development targeting rural development should always consider localization of the user interfaces to their linguistic and cultural needs for interface usability improvement?

- A. Strongly Agree
- B. Agree
- C. Not Sure
- D. Disagree
- E. Strongly Disagree

17. After using the demonstrated system, did you better understand how to use the interfaces and would you be able to work on your own without any assistance?

- A. Strongly Agree
- B. Agree
- C. Not Sure
- D. Disagree
- E. Strongly Disagree