39th Inaugural Lecture

"Software System in Mother Tongue: The Yoruba Perspective"

by
Prof. Oladokun Olopade

Preamble

I welcome you all to the 39th Inaugural Lecture of Lagos State University. It is a great honour for me to stand before this distinguished audience to deliver the 3rd inaugural lecture of the 2008/2009 session of this great University. It is the fourth from the Faculty of Engineering and the second from the Department of Electronics and Computer Engineering. The topic of the inaugural lecture is "Software System in Mother Tongue: The Yoruba Perspective".

Let me confess from the start that I have always been an apostle of mother tongue in all aspects of life. Looking back, I can still recall that I always liked to do things in accordance with Yoruba culture and tradition and, most especially, expressing myself in the Yoruba language. Also, I must confess that I never liked the English language and I still don't cherish it, but I realized that for me to succeed in this environment of my birth, I must be proficient in this foreign language called English.

Let me, at this stage, give you just two examples to show that I am not comfortable with foreign languages. The first example. I was born into a poor illiterate Muslim family and my "Muslim" or to be more correct Arabic name is Zakariyau although I was being called Saka. As a young boy, there was that something in me that was pushing me repeatedly to ask my parents for the meaning of my name. The responses were varied and I was not satisfied. Fortunately, when I was in form one in Mayflower School, Ikenne I got hold of the Holy Qur'an in Yoruba through an uncle of mine. I read through the relevant portion and I was lucky to know the meaning of Zakariyau as "Akó Ọlọ́run Lékàn." What a joy. I went to my father and told him of my discovery. Thereafter. I requested that I would like to be called "Akó Ọlọ́run Lékàn.". My father's face turned red in anger. I was sent out of the house, and it was an uncle who rescued the situation. I stood by my decision and changed Saka to Lékàn on all my documents. It was when I was admitted to the University that my father finally forgave me.

The second episode was also religious. It was an encounter with my British principal in Egbado College, Ilaro. I was with him as a prefect to conduct the morning devotion. To end the prayer the principal called us to say the Lord's prayer. I recited it in Yoruba "Baba wa tin be ni orun ….." while he did his in English. Immediately after the Grace, he told me to follow him to his office.

There he accused me of speaking in the vernacular, a very grievous offence in the college. I told him I was speaking to my God. He took my reply as rudeness. He then warned me that if I repeated the offence, he would demote me as a prefect.

My sin in these two cases was that I was expressing the content of my inner mind in my mother tongue. I briefly attended Qur'anic school with joy, and all my education had been in English but I never thought in it. My situation is best described by Karl Marx when he said:

> *The beginner who has learned a new language always translates it back into his mother tongue, but he assimilates the spirit of the new language and expresses himself freely in it only when he moves in it without recalling the old and when he forgets his native tongue.*

Since vast majority of us have not forgotten our mother tongues can we neglect them and make progress when we know that language is the foundational tool of human development?

What is Mother Tongue?

Mother tongue, also known as native language or first language, is the language of the community into which one is born and lives. Let us consider the case of Mr. Tokunbo, of Yoruba-Nigerian parentage, who was born and bred in England. His mother tongue will be the English language because of his total exposure to

English culture. Another case is that of Mr. Peter of British parentage born in Ibadan. His parents are lecturers in the University of Ibadan. They lived within the elite community of the University. Peter's mother tongue will be the English language. When a person is monolingual then that only language becomes his/her mother tongue. For another person that is multi-lingual, the language in which he/she thinks most and speaks most can be called his/her mother tongue, based on Karl Marx's submission. In a monolingual environment the only language spoken is the mother tongue, while in a multi-lingual environment, the strongest language is the mother tongue which is considered to be the best vehicle used to express one's thoughts.

The Human Languages and Thought

The common sense view seems to be that the function of human language is communication, it does more, language expresses thoughts that are already there prior to their expression. Because language transcends the individual in favour of his community, it consequently becomes our property and that of our culture. It is through language that we acquire and transmit our knowledge and our know-how which facilitate certain domination over our environment. It stands indeed as the key component and the barometer of our development. Are the G8 not dominating the world through their mono-lingual cultures? The relationship between language and thought becomes complex in a multi-lingual

environment. The question then is, How efficiently and effectively can a speaker in this environment express his/her thought?

The answer can be found in the Sapir-Whorf hypothesis which states:

> *We are thus introduced to a new principle of relativity, which holds that all observations are not led by the same physical evidence to the same picture of the universe, unless their linguistic background are similar, or can in someway be calibrated.*

This, in short, means that different language patterns yield different pattern of thoughts. A corollary of this hypothesis and results of experiments conducted by other researchers, especially those who belong to the school of linguistic universality, support the idea that the strongest language of a multi-lingual speaker dominates or is dominated by thought. Whether language controls thought or thought control language is ultimately unprovable.

A summary of their findings that are relevant to mother tongue application is as follows:

(1) Human language and thought are inseparable

(2) Human language is rooted and embedded in culture

(3) The strongest language of a multi-lingual speaker dominates his/her thought.

We can therefore say that when a person uses his/her mother tongue, it is found that complex ideas might just be easier to grasp and that the level of understanding of the same idea in a second language is generally lower or at best can be at par with that of the mother tongue. It is therefore not surprising that some non-English-speaking programmers are now exploring the possibility of using their mother tongues to create and optimize the three core tools to software programming philosophy. That is, they are using their mother tongues to:

(a) write program as symbolic textual and graphical artifacts

(b) specify software requirements (the whats) separately from the program (the how) and the ability to pit one against the other.

(c) Structure the behaviour according to the system's structure, providing each piece or object with its full behaviour.

For example, European countries and Japan have already taken and are taking steps along the above approach with success. Of most important is the developing countries of the world. They must adopt this strategy if more than 80% of their population that can neither read nor write in English, Chinese or Spanish will ever benefit from the dividends of ICT. The development and application of software packages in their various native languages have definitely become part of the major instruments of an all inclusive development they must pay serious attention to.

The development and application of software packages in their respective mother tongue will promote multilingualism and ensure the continuous existence of their languages alongside other international languages. It must be emphasized, at this stage, that the mother tongue software based initiative has to be championed by all concerned shareholders, from the individual to government, for it to succeed. I do not want to be labeled as an alarmist but the prognosis is that any language that fails to join those being used as tools for software development and application is likely to become extinct before the end of the millennium.

Software Definition

Today, computer software is the single most important technological product in the world. It is both a product and a vehicle for delivering other products. As a product, for example, your cellular phone is a computer system, and as a vehicle the operating system acts as the basis for the control of computer. Software consists of not just codes in machine readable form, but also all documentation that is an intrinsic component of every project. Also, software includes the specification document, the design document, legal and accounting documents, the software project management plan and other management documents, as well as all types of manuals. In short,

Software = Programs + Data + Documents

The programming package is considered as a major unit around which other units are built. When a software system is developed, its core (program) falls into one or a combination of these broad sub-package categories:

(1) System

(2) Application

(3) Scientific/Engineering

(4) Embedded

(5) Product

(6) Web Application and

(7) Artificial Intelligence.

A software package can further be classified along product development as being

(a) Object-oriented paradigm

(b) Classical (traditional) and

(c) Logic Paradigm.

It is to be noted, however, that the majority of experts in computer engineering and computer science are silent about classifying software packages on the basis of the natural languages used to develop them. This omission is pardonable because the vast majority of experts in these areas are Americans, British, English speaking

Canadians and Australians whose mother tongue is the English language. Today, software system categorization along natural language is very important if the impact of ICT is to be felt by half of the world's population that are non English speaking.

Software and Human Culture

Software applications are designed around user's interaction. The users of a software package interact with it through a user's interface. This interface then controls the interaction of the users with the applications' core functions of the package. Let us assume that we have a package where we have solved all technical issues and a properly compiled program is ready, do you know that at execution the outcome may be far from what we expect?. This situation should remind us of the old adage that "building the system right" is not the same as "building the right system". Research, especially that related to application of software packages, has shown that this type of "failure" can be traced to the presence of embedded cultural assumptions in the software package. This assumption may seem alien or even inimical to the culture of the user. For example, an HR package from the U.S that is based on American culture, say, "one man-one wife" may not produce a good management report if installed in Nigeria without necessary modification (editing).

Research has shown that:

1. Software technology is rooted in and shaped by culture
2. Unlike many other technologies software describes and automates complex activities and whole processes that previously were undertaken by people and organizations.
3. The core of a software artifact embeds decision-making, rules of behaviour and patterns of actions that depend on culture

Here the culture we are referring to is embedded in the language and other symbols that can be interpreted as language. The software developer then uses his or her language to develop the package and in so doing, naturally stamps his or her culture on the software package. Research shows that the culture of the developers is less pronounced in mathematical and scientific packages. This is because mathematical functions and scientific symbols have been internationalized and so made common to all cultures. However cultures can easily be noticed in software packages used to perform the activities of human beings such as in e-commerce, e-government and web applications. Then our next question is, whose culture do we see in existing software packages? As of today, a large majority of world's software are in English and they are developed by nationals of four primarily

English–speaking countries namely United State of America, United Kingdom, Canada and Australia.

Statistics shows that of 8,500 programming languages in the world, U.S.A developed 2,400, U.K, 600, Canada 160 and Australia 75, and all these are in the English language. The U.S.A further dominates the production of operating systems like Windows, Unix, the most widely-used word processing, hypertext, spreadsheet, multimedia and Web packages and they are all in English. The Internet and Web are not left out of English domination. All efforts by Japanese and European software developers and manufacturers to challenge English domination in production of software have so far not been successful (Kenishm 1997). Although there are a few notable exceptions like SAP, a German based package yet a German program and non-German users would still prefer to work from English to use this package written by Germans. The major reason for this is that English has become the best preferred second language for most non-English ICT professionals and users. In summary, most of the software in their original forms are culture dependent and because about 90% of dominant packages originate from the US, we can safely conclude that the world of computing is being controlled by American-English culture. It is to be noted that despite the fact that English is the mother tongue in the big four countries there are still noticeable cultural differences in packages developed by their nationals.

Existing Software Platforms in Non-English Speaking Countries

History tells us that the invention and commercial production of the modern computer system and its peripherals began in the English speaking countries of the world. Therefore, these countries have not committed any sin for developing software packages that will complement and supplement the system in their one and only one God given mother tongue - English.

It is therefore natural that all programming languages developed in these four countries are written in the English language. These packages are so popular that English–based programming languages now almost monopolize the software market worldwide. For the non-English speaking countries to be relevant in the ICT revolution, they must embrace the use, and massively too, of software packages. How then will they do this? They must do one or a combination of the following:

1. Produce software packages in their own languages
2. Take the advantage of existing English-based packages and localize them

Before we proceed, let us recognize the two broad groups of countries under the non-English-speaking countries. The first group is made up of countries in Europe (except U.K), Japan and China. The second group is made up of developing countries in Africa, South America, Asia and the Middle East. Countries in these

groups are combining all available schemes to remain relevant in the software development project. Individuals, corporations and educational institutions are going into the development of software packages based on local languages. The result is that today, there is a distribution of about 0.5 programming per nation. This cannot be compared with 600 programs developed in the U.K. Examples of non-English programming languages developed are:

Spanish - Lexico

Russian - Rapira

Japanese - Ruby

Chinese - Basic Chinese

Arabic - Arlogo.

The success of English–based programming language and the ascendancy of English as a global language cannot be wished away. So there is a large increase of nationals from group **1** who are studying English as a second language so that they can use it for the application of English based packages.

The second option which is the localization process is the most popular. Most of the packages developed by the Americans before 1990 were on the basis that they were meant for only English speaking communities and so there was no internal structure in the programming language packages to support smooth and orderly

translations. Some of the translations of pre 1990 packages that were embarked upon either failed or the translation became a re-engineering of the package. From the early 1990s, the American manufacturers and target users in Europe met at various locations and agreed to work together on localization of viable packages. Funds were made available for the project. The American software firms restricted the coding of the packages and since then localization had become big business bringing millions of dollars to American companies. Today almost all English based programming packages are localized for European applications. A similar approach was adopted in China and Japan.

The situation in the developing countries was very different. Most of them were colonies of European countries. The infrastructures in these countries were not developed, and a vast majority of the people were uneducated. In fact, there was no basis for these countries to think of development and/or application of software packages in any languages in the 1990s. Then some of their local languages were disappearing while others were struggling to reach maturity. The only exception was India. There were a few active nodes in India where serious activities were going on the development of indigenous Indian programming languages. Today, the production of Swaram is to their credit. One would have expected that the localization process would thrive in India as in Europe but there are two hurdles. The first is that there are many local languages in India and the big American

software companies were not sure of the language or languages that would bring good return on investment. The other problem is that the Indian scripts are not based on Roman alphabets and so it will be transliteration and not translation.

The other countries belonging to group 2, are made up of developing nations. Nigeria can be taken as a typical example. Most of the languages in Nigeria lack strong technological and political base for development. The number of speakers is low for economic consideration. Where the population is high enough to support economic activity the majority of the population is found to be uneducated. If, however, one can identify a group whose language can be used, the only option is to adopt the localization process. This has started in South Africa where eleven languages have been identified and Microsoft office suite has been localized in these languages. Also now Microsoft office package has been released in three main Nigerian languages, namely Hausa, Igbo and Yoruba. .

Need for Localization

Users of localized softwares must always expect the software packages to interact with them in their native language. The reason for this is simple. It makes the software more captivating, and user friendly, and the users find it very easy to grasp its technical details. In this way, the overall level of adaptability and productivity of people of an organization incredibly increases. Because the software talks the native language of its users, the chances of mistakes decrease,

and the chances that the mistakes will be corrected quickly are high. However, not only should the language of the software be the familiar one, the pictures, logos, interactive windows, graphical modules etc. should also conform to the cultural values and ethics of the users in a particular region. The software in any case, should not hurt the emotions of its users and must not discriminate on the basis of caste, colour, gender, creed, religion etc. It is possible that a particular image, phrase or even design which is considered as good in a particular region is not acceptable in other places. Thus, during the process of software localization these factors are also considered and apart from the language translation certain components are also changed, if necessary, keeping in mind the targeted users.

Characteristics of a Good Localized Software

Some of the characteristics of a good localized software could be summarized as:

(1) The end users should be able to interact with the software naturally

(2) Users can use their standard keyboard without any difficulty and should be able to enter the data in the same way such data would have been written in their mother tongue.

(3) Any type of messages which may be flashed to the software must be comprehensible and must reflect that particular regional touch.

(4) The localization is not restricted to the software only, but it should also be done for any kind of documents associated with the software which may be in the form of soft copies or in printed materials.

(5) The software localized should conform to the cultural values and ethnics of the users in a particular region. The software, in any case, should not hurt the sensibilities of its users, and must not discriminate against anybody on the basis of caste, colour, gender, creed, religion etc.

Software Localization Process

In order to translate a software from one culture to another, the first and foremost requirement is to get hold of its source code. Then all the required modifications are done in the source code keeping in mind the target users. It is to be noted, however, that localization was initially approached by American software firms as an "add-on"; i.e., after the original program was fully functional in English, "localizers" were put to work to produce, for example, Spanish, French, Japanese versions. The localized outputs were found to be sometimes re-writing of the source code. It was then discovered that the best approach was to assume, at the developmental stage, that every package would need to be localized. Therefore, programmers were advised at the design stage to determine and separate culture

dependent components from culture independent components. This process is known as internationalization, and is the first step in software localization. The culturally dependent and independent components are separately translated into the language of the target community (the users). In the target community, the culture dependent components are bundled to become the user interface which is coupled with the application core that stores the culture independent components. The final step involves testing the application's functionality and the user interface and conducting a linguistic review. If additional changes are needed, all team members must verify them before implementation.

The importance of this localization process can be seen as major firms like Microsoft(MS) currently release dozens of language versions of new (or upgraded) operating systems and applications are struggling to do more. For example, with this approach, MS released her office suite in eleven (11) South African languages in 2007 and, in the last quarter of 2008, repeated the same for Igbo, Hausa and Yoruba versions.

Key Elements to be Localized

The following are the major key elements to be localized:

1. Resources Files (objects like menus, dialog boxes window frame, bitmaps, and cursor shapes and their respective text-content)

2. Help files (help guides, footnotes, hyperlinks, search options, glossary etc).
3. Instruction manuals (all documents)
4. Screenshots and Bitmap Images (such as splash screen, message box, and command buttons)
5. Index localization

Challenges in the Software Localization Process

Localization is meant to be in two directions, but today the majority of localization process is that of products from the advanced countries to the less developed communities of the world. The smaller receiving communities are found to be more tolerant of foreign language elements in the localized product. This tends to have special effect on the future of their language and cultures. Therefore, localization planners must ensure that the range of translation strategies is restricted by the nature of the target culture involved. This, therefore, calls for the involvement of highly proficient local linguists at all stages of the process, otherwise, for example, in Yoruba communities the localized product will be half English and half Yoruba.

Is there an Alternative to Localization?

Ideally, every community with her local language is expected to behave like the Americans and develop software programming packages in their God given

mother tongue (first language). So, the alternative to localization is package re-engineering.

However, for a community to embark on re-engineering as the Americans (USA) engineered software development that nation must not only have in place appropriate infrastructures but also the technical, economic and political wherewithal to succeed. Even Europe that is part of the developed world can only boast of developing less than 100 of non-English software packages. The bold attempt by Indian software experts to produce software programming packages in India languages is facing a lot of problems. However the success of localization of English based software packages in India is by far more effective and pronounced on the population than that of the three re-engineered packages in Indian languages. There are a few nations that have produced one or two packages each directly from scratch. These successes are well acknowledged. For example, Rugby is very popular in Japan, yet its successes cannot match those of localized packages. We can therefore conclude that although the alternative is package re-engineering, but it is economically unprofitable; thus, for sometime to come, localization process will continue to be adopted to satisfy the software needs of non-English speaking countries.

Software Development in Yoruba

I will like at this point to recount some points raised earlier in this lecture, which I consider as foundational issues in Yoruba software development. I attempted to summarize the findings of the linguists in which we were informed that language and thought are like egg and chicken, whereby one controls the other. It was also shown that no language is superior to the other, and that mother tongue is the best vehicle of expressing one's thought. These have helped us to dismiss the notion that African languages cannot express new and emerging ideas and concepts, probably because the cultures of the African speakers do not include such ideas and concepts. It can be inferred that because the big four are mono-lingual and share the same mother tongue contribute immensely to their supremacy in the field of software development. The other countries like Japan, China, France and Germany that equally succeeded in this field also embraced their mother tongues under the process of localization.

What does all this information mean to people in non-English speaking and developing nations in the world and, in particular, to the Yoruba community in Nigeria? We are challenged. If it is therefore the wish of every average Yoruba man or woman who can neither read nor write in the English language wishes to be digitally compliant, he or she is expected, as a computer user, to be able to do the following:

(a) operate a word processor in Yoruba

(b) learn to solve problems using software packages written in Yoruba
(c) know how to use the World Wide Web to obtain information on whatever interests him or her.

These three broad desires can only be satisfied when Yoruba users have in their computer systems tools that can handle the extra characters. The solution is the provision of a Unicode encoding scheme for Yoruba orthography and a Yoruba keyboard.

The Yoruba Language

Yoruba is one of the four official languages of Nigeria. It is spoken by about 30 million people s with 21 million of them living in the original homeland in South West Nigeria. The rest are in Benin Republic, Togo, U.K, U.S.A and Brazil etc. Yoruba is the 29th largest spoken language in the wrld and the first in Africa. It is considered as a minority language but it has matured orthography. It had been used to translate the English Bible to the Yoruba Bible as early as 1850. In its written form, modern Yoruba uses the Roman alphabet. It has seven oral vowels (a,e,e,i,o,o,u), five nasal vowels (an, en in on un), and three tones, the acute low tone (\ ), the middle tone ( ) and the grave high tone (/). Yoruba orthography does not use the letters c, q, v, x and z. Because of the additional diacritic to the standard Roman alphabet, written information cannot be exchanged between the well-known computer systems. The reason for this is simple. The computer

systems were designed mainly to take care of the languages used only in the American and European markets. The solution, therefore is to identify appropriate Yoruba fonts, define the requirements for its code and then design a standard Yoruba keyboard to ensure correct transfer of data.

Various committees of voluntary experts, private software organizations, computer science departments of universities and individuals networked, exchanged ideas on how to solve these problems for Yoruba users of computer systems. The Internet and Web were of great importance by providing necessary platforms for exchange of ideas. Today, as a result of collaborative efforts of experts in computer science, linguistics and the software industries, the Unicode CLDR for Yoruba has been submitted and approved by Unicode Consortium. That means each Yoruba alphabet now has a unique location in the 65,000 places provided by Unicode consortium for all the alphabets and characters used for writing in the entire word. Also Unicode, which contains local (cultural) data called locales for 127 languages, 140 territories and 374 locales has a page for Yoruba, where all special features about Yoruba language and locales are stored.

The next challenge is to have all major operating systems like Windows, Linux, and OS2 installed in our computers and create a new codepage that has a collection of Yoruba letters. The successful encoding of Yoruba characters and the generation of Yoruba locales were not a simple task. It took the experts many

years of consultation. We must acknowledge their voluntary contributions. Similar big projects like this in many countries were recognized as been strategic and therefore supported by governments. For example, the governments of India, Sir Lanka and South Africa provided funds for a similar project. I salute Toyin Falola, Department of History, The University of Texas, Dr. Tunde Adegbola of African Language Technology–Initiatives, A.M. Onibudo, Prof Kola Owolabi, Dr. S. K. Olamijulo, and others. It is worthy of note that there were some non-Yoruba who actively participated in the preparation of the glossaries. We commend the contributions of Messrs Alberto Escudero, Pascal and Louise Berthilson and Don Osborn.

Yoruba Keyboard

Let it be known that the present computer keyboard (hardware and software) is not designed for the Yoruba language. Yoruba is a tonal language with seven focal vowels and three nasal vowels, so a Yoruba keyboard in its minimum format will need 30 keys to accommodate all the vowels. For its orthographic needs another key is required to take care of ***s*** with a dot under it. It is possible to design and produce such a keyboard. However, software engineers and scientists have developed other approaches for the production of a keyboard with a Yoruba character set that will satisfy both its phonetic and orthographic needs with

varied accuracy. For the keyboard to be ergonomic and efficient, most researchers are working on the modification of the QWERTY layout by re-arranging the positions of the characters and the development of appropriate keyboard software to drive it.

Research work along this line has been going for the past 20 years, and some prototypes have been produced with varied degree of success. Listed below are some prominent prototypes that have been produced, tested and put into use:

1. Konyin keyboard
2. Wazobia keyboard
3. NITDA keyboard
4. ABD keyboard
5. Alt-I keyboard Version 1
6. Alt –I keyboard Version 2

The first five listed Keyboards have been around for some years going through various applications. They are all modified QWERTY in the physical but driven by different softwares. Although each one of them has its limitations, yet users are forced by lack of better alternatives to use them. Some of their shortcomings are:

a. User may sometime need to make up to five keystrokes to produce a character (e.g. ọ́)

b. The character produced may have its diacritics not properly positioned.

c. The character produced may shrink.

d. It is cumbersome to manipulate the combination of special keys like Altgr, dead key, shift, option etc in producing a character.

I am happy to report that all of these problems and others not listed have been solved with the release of a new prototype keyboard from African Language Technology Initiative. The organization has completely re-designed their first keyboard based on Unicode scheme.

The new Alt-I keyboard ver 2 takes the advantage of the fact that written Yoruba never uses the letter Q, Z, X, C and V and so exclude them from the layout. The font was based on Unicode scheme, while the layout was based on statistical distribution of characters. All these are part of the standard interface package for mapping of full Yoruba characters in Unicode set to drive the keyboard. The new keyboard has been tested and found to be very efficient. It takes a maximum of two key strokes to produce any character. It was reported that this Alt-I ver 2 Yoruba keyboard has not only solved most of the problems facing Yoruba users but it has opened up research avenue to solving problems associated with other tonal languages in Africa. This keyboard won the year 2008 ITCD. Award on Local Content Applications of Information Communication Technologies (ICT). The researchers were highly commended for solving a major problem in the

application of software in Africa. The award is one of the Africa Information Society Initiative (IASI) Media Awards of United Nations Economic Committee for Africa (UNECA). Now with Alt-I ver. 2, the Yoruba language has a keyboard that can perform on various computing platforms. For now this keyboard is not available commercially, but its developers have approached big software companies like Microsoft to help integrate it with its MS word. The other good news is that Microsoft released Yoruba version of MS office in the fourth quarter of 2008. It will therefore be very good business if the next release of MS office Yoruba version comes with Alt-I version 2.

There are 30 million prospective users waiting for this keyboard. We must add that African Language Technology-Initiative is about to release another product, Yoruba text-to-speech software package. When this is available it will help users.

I wish to acknowledge the effort of Dr. Tunde Adegbola and his team at African Language Technology-Initiative for their contribution in providing these tools that will accelerate software localization and re-engineering process.

EXAMPLES OF LOCALIZATION PROCESS

With the availability of a standard interface package for mapping of full Yoruba characters, character combinations and their input in ASCII and Unicode and (b) Yoruba keyboard we can now consider how they are applied in software development.

Example 1: Basic program localized into Yoruba Language.

Please find below a simple basic program written in the English language and its translation in the Yoruba Language.

10	Y = 0	10	Y = 0
20	INPUT E,	20	ÌGBÉWỌLÉ E, F
30	FOR A = E TO F	30	FÚN A = E SÍ F
40	Y = Y + LOG(A)	40	Y = Y + LOG(A)
50	NEXT A	50	MÚTUNTUN A
60	PRINT Y	60	TẸ̀JÁDE Y

This translation could have been written 20 years ago. But then that program in Yoruba would have been viewed as cabbage by any computer system and the output would have been cabbage also. The reasons are:

1. The computer system then and up to year 2004 could not recognize Yoruba alphabets.
2. As late as year 2000, there was no computer input device (keyboard) that could be used to produce Yoruba characters and transfer them to the processor of a computer in internationally recognized format.

These and other related problems have been researched into and a review of the solutions to them are the major subjects of this lecture. We are happy to report that we now have the Yoruba characters recognized under the Unicode encoding

scheme and there are standard keyboards for the Yoruba language. These are the two basic tools needed as interface to execute any Yoruba language based program.

In the process of localization, the localizers must recognize that four different languages are employed. These are:

1. The language of the source community (the base Language the developers use). In this example the English language.
2. The programming language: This is the combination of the methodology and structure used in the coding of the program. It is this that gives the distinction among the programming language (Fortran, C. Jova Basic etc). The example here is Basic program.
3. The mathematical language: This contains all mathematical symbols (+, - + / etc) and functions like sine, log etc. these are internationalized and are not translated.
4. The language of the target community. In this case, Yoruba.

It is the recognition of these in any given source program and their proper translation that constitute localization, all other documents like user manuals must be translated into target language.

EXAMPLE 2: OFFICE AUTOMATION APPLICATION

Prior to the fourth quarter of 2008 when the second version of Alt-I Yoruba keyboard was released desperate users were using keyboards driven by softwares that were proprietary in nature and therefore not cross – compatible. This was because different developers had used different ASCII codes to represent the letters of the Yoruba language that are not shared with the English language. Such a keyboard was used to type the Yoruba language version of the Basic program in example 1. With the release of Alt-I version 2 in 2007, which is compatible with Unicode standard, Yoruba text cross compatibility is assured. At the tail end of 2008, this was complemented with the release of a Yoruba translation of Microsoft Windows Vista and Microsoft Office 2007. With this development, any person who can read and write in the Yoruba language will be able to use Microsoft Word, Excel, Access and Powerpoint just in the same manner any literate English speaking user will use the MS Office.

Let us review a display of a window under a Yoruba platform. The application of Microsoft Word package of MS Office 2007 with its Yoruba language localization scheme begins with the display of a Window titled Ọrọ Microsoft. If the user then clicks the office button on the upper left hand corner of the Window what will be displayed on the screen are commands given in Yoruba language as listed below:

- Ojú ìwé tuntun instead of New

☐	Sí	instead of Open
☐	Fi pamọ́	instead of Save
☐	Fi pamọ́ gẹ́gẹ́bi	instead of Save as
☐	Tè	instead of Print
☐	Gbaradi	instead of Prepare
☐	Fi sọwọ́	instead of Send
☐	Tẹ́ jáde	instead of Publish
☐	Pa dé	instead of Closeo

The messages on display are composed of simple and basic Yoruba words. They are easy to understand and support quick interaction with the software package.

Example 3: Web Page Application

In 2005, Google announced the inclusion of the Yoruba language as an option for use in its search engine. This is the first Nigerian language translation on the Web service. This will afford the Yoruba speakers in the country the opportunity to access and post needed information via the medium in the Yoruba language. Now, an Ekiti farmer who can only speak Yoruba, when equipped with an appropriate Yoruba keyboard, can advertise his products on the Net and receive responses from Yoruba speakers who have access to the Net. For now, however, if you query the Net in Yoruba you are likely to get most of your response in

English and very few in Yoruba. This is because there are very few Yoruba websites. What is therefore needed now is for Yoruba speakers to create Yoruba websites.

Contribution by the Author

The inaugural lecturer has made some modest contributions to the development of technology. Some of them are outlined below:

Development of very fast Protective Scheme for EHV Transmission Lines

In the 70's the fastest protective relay was taking about 20ms to detect and clear faults on 132KV transmission lines. The General Electric Company of England funded a research work in the department of Electrical Engineering, University of Manchester Institute of Science and Technology to improve on the relay operations. This project became my PhD research work. At the end, I developed a computer relaying software package that reduced the operation time to less than 10ms. This was the fastest in the world then. The package and modified versions are incorporated in some digital relays manufactured by G.E.C.

Yoruba Talking Drum Technology

One of the research interests of the inaugural lecturer was the study of the behaviour of Yoruba talking drums. I was able to model the talking drum as an electrical circuit and from there extract all its major characteristics. This will definitely be of use when the skin and special wood for making the drums are no more available. Then users will be able to use computer to generate equivalent sounds.

Application of Popular Software Packages in Teaching Electrical Engineering Courses.

Because of his deep knowledge of computer system and good understanding of the Yoruba language, the author had for years in his teaching profession adapted and adopted the Word, Excel and Access of Microsoft Office to teach courses like circuit theory, analogue signal and systems and digital electronics, especially when packages like Matlab and PSpice were not available. For example, Microsoft Excel can easily be manipulated to demonstrate the behaviour of gates, flip-flops, decoders etc.

Yoruba and Technology

The submission that African languages cannot express new and emerging ideas, notions and concepts, probably because the cultures of African speakers do not include such ideas and concepts, is false. The people who are the propagators of this hypothesis do not believe that there can be new concepts and notions in Yoruba, but only in foreign languages like English. The problem is with us the educated elite who can only explain new concepts in English using English. Those of us who have decided to use Yoruba to explain the so-called new concepts had discovered that our audience do understand whatever concept the new word conveys. Let me remind you that Yoruba became a matured language in 1850

because the Holy Bible was translated into Yoruba that year. The Bible as a book contains almost all the concepts and notions we have in the world. It is therefore an achievement on our part to make use of the Yoruba language in our teaching, especially in technological subjects.

RECOMMENDATION

1. The Nigerian ICT4D policy should strongly support the use of Free and Open source software (FOSS) packages in the three regional languages in its first phase, and later bring in more local languages.
2. Mother tongue should be made the language of instruction from primary school to secondary school.
3. All ICT products providers should be forced to localize their products
4. Positive attitudes about mother tongue among educated Africans should be promoted.
5. Since we now have MS office suite in the three official regional languages (Igbo, Hausa and Yoruba) it should be made mandatory that all information from government, like e-governance products, should be published in these languages
6. All new procurement of software packages in the country should demand that these products be localized.

7. Curricula of departments of computer science and engineering in Nigerian higher institutions should be reviewed so as to include the development, deployment and application of packages based on mother tongues.

Conclusion

Mr. Vice-Chancellor, Sir, let me conclude this lecture by reminding the audience that there is a cognitive basis for better programming language acquisition when the developer uses his or her mother tongue. It allows for a deeper understanding of the schema behind the computer programming. Computer software packages in the mother tongue will increase work productivity faster because terminology in the native language will not bring jargons. Sir, despite the fact that the English language has unofficially become the lingua franca of the world we must recognize that Yoruba language is the key for development in the Yoruba nation.

ACKNOWLEDGEMENT

Mr. Vice-Chancellor, Sir, let me now move to the most difficult task of expressing gratitude to all those who have contributed to my being the before you today. First, I give thanks to God Who knows that I do not have enough words to thank Him. Baba Eledumare e se o.

The work presented in this lecturer is not restricted to the author. It is a summary of works of great people and they all deserve to be acknowledged. I wish to acknowledge some, and these include Dr. Tunde Adegbola, Dr. Odetunji Odejobi, Dr. Onibudo, Prof Francis Egbokhare, Prof. Olu Longe, Prof Akinwumi Ishola, Prof Tunde Adeniran, Prof. Kola Owolabi and our great mentor, Prof. Babs Fafunwa.

Next, I would like to thank the Vice-Chancellor for giving me the opportunity to give this inaugural lecture. I always enjoy your contributions in Yoruba. I acknowledge your effort in building a new LASU and I thank the Dean of Engineering for the actualization of a new LASU in Epe.

My thanks go to my parents (of blessed memory). I cannot forget how they were borrowing and selling their belongings to see me through primary and secondary education. I cannot forget my sister, Mrs. Titilayo Adepoju, who was denied education for me to be educated. I owe an immeasurable debt of love and

gratitude to my late aunt, Mrs. Olatundun Oladejo, who was with me on my first day in the University of Lagos and witnessed how I could not gain access to my room for failure to pay accommodation fees. She wept at the gate of Hall 1, We both returned to Ibadan that night dejected. She went to sell almost everything she had to pay my first term's fees. I must appreciate Engr. Ladi Osinowo and Engr G. Fashina for how the three of us were using their two meal tickets for two terms because my parents could pay for my tuition and accommodation

I will ever be grateful to Prof. Arvind Shah, my supervisor in the Department of Applied Physics, Swiss Institute of Technology, Zurich, Switzerland. He encouraged and supported me in my first ever scientific project. The practical experience I gained in the building of a working micro-computer from scratch using only basic components has been my great strength in both academic research and teaching.

In LASU, I wish to acknowledge the cooperation I had with late Prof. Debo Falade, He was an excellent manager. I also wish to recognize my very great friend Prof. S.A. Adekola, He was and still dishing out to me those important pieces of advice.

Finally, I come to thank my nuclear family: first my children: Mrs. and Mr. Foluke Olubusi, Mr. and Mrs. Yomi Olopade, Mr. and Mrs. Loba Olopade for

your love, advice and support. I acknowledge your technical contribution to this lecture. I thank my brothers, sisters, aunts, uncle, for various kind of support that I got from cradle till date.

Finally, Mr. Vice Chancellor, Sir, this is the time the truth must be told. The person that we should be celebrating today is not the lecturer, but that person that I met in 1968. I am talking of that God-sent angel in my life. As I stand before you, I don't know the size of my shoes or the length of my trousers; I don't know how food gets to the table; in fact, I don't know where my certificates are. The person that cares and knows all about me deserves to be honoured. Please, I acknowledge the efforts of Mrs Bolanle Atinuke Olopade not only on my life but also on the entire family; I thank God for your life for bringing me to this height.

And to all of you that are present, please accept my thanks with love. Thank you.

REFERENCES

1. **Butzkamm, W. (2003),** We only learn Language once, The role of the Mother tongue in FL classrooms: death of a dogma, Language Learning Journal, Volume 28, Issue 1, pp. 29-39, Routledge, Winter.
2. **Souya, T., Hayakawa, E., Honma, M., Fakushima, H., Namiki, N., Takahasi, N., and Nakagawa, M., (1991).** Programming in a mother tongue: philosophy, implementation, practice and effect. Proceedings of the fifteen Annual International Computer Software and Application Conference, pp. 705 – 712.
3. **Choudhary, A. (2008).** Mother tongue based programming Languages. http://www.mail-archive.com/llugd@lists.linux-delhi.org.
4. **Haugen, E. (1996).** Death of a mother tongue, New Scientist Tech.
5. **Kersten, G. E, Kersten, M. A., and Rakowski., W.M. (2002).** Journal of Global Information Management, 10(4), pp. 86-10.
6. **Keniston, K., (1997).** Software localization: Notes on technology and culture, program in Science, Technology, and Society, Massachusetts Institute of Technology.
7. **Folorunso, O. (2006).** The famished artist in a famished society, 27th Inaugural, Lagos State University.
8. **Ajose S. O. (2008).** Electrical Engineering: Key to development, Inauguard lecture, Lagos State University Ojo Lagos.
9. **Olopade, O. L. (1972).** Design and building of a 4-bit microcomputer, Institute for Technology Physic an development ETH Zurich.
10. **Olopade O. L. (1975).** Interfacing and communication problems in computer system, Msc Thesis University of Manchester Institute of Science and Technology, Manchester.
11. **Olopade, O. L. (1978).** A High speed protection algorithm for electrical power lines. Ph. D. thesis University of Manchester Institute of Institute of Science and Technology, Manchester.
12. **Olopade, O. L. (1980).** "Development in the calculation of wave forms spectra for Transient fault currents and voltages". IFE proceedings. Vol. 127 part C. No. 3, May 1980.
13. **Olopade, O. L. (2002).** Evaluation of formant frequencies of African memberano-phonic drums' Journal of Applied Science and Technology, Faculty of Engineering, 2002.
14. **Olopade, O. L. (2002).** Derivation of an acoustic equivalent acoustic network of a circular drum membrane', Journal of Applied Science and Technology.

15. **Olopade, O. L. (2002).** A high Speed distance Protection Algorithm for EHV Transmission Lines', Journal of Scientific Research and Development.
16. **Olopade, O. L. (2002).** Symbolic Computations in Electrical and Electronic Engineering', Journal of Engineering Research.
17. **Olopade, O. L., (1987).** The electronics revolution strategy for effective application in Nigeria' (National conference on science and technology, University of Ibadan).
18. **Olopade, O. L., (1985).** 'Innovations in computer technology: Relevance in the West African Sub- Region', (West African Management Development Institute Network (WAMDEVIN) Forum, Lagos, Nigeria.
19. **Olopade, O. L., (1992).** 'Hardware and Software Requirements including Networking for an Intergrated Population and Development MIS', National Workshop on Integrated Population and Development Management Information System: Conceptual and Methodological Issues: Federal Ministry of Health, Lagos, 25^{th} – 27^{th} Feb.