"A Study on the Approach of Architecture Reference Model for Design of Multilingual Applications"

Ganesh Babun[1] Dr. Mahammad. K. B.[2]

[1]Research Scholar, CMJ University, Shillong, Meghalaya

[2]Professor of Computer Science, CMJ University, Shillong, Meghalaya

Abstract – Multilingual software development is put in limelight due to the globalization efforts of the organization. But these development approaches do not have any reference model or framework from the language perspective. This situation limits multilingual software development. In order to address this, we developed architectural reference model for multilingual software (ARMMS) using unit operation. This model ensures the qualities like maintainability, understandability, reusability, adaptability and language neutrality. Also, we present views of ARMMS. Furthermore, a discussion is made on relating unit operation and above said non-functional qualities of multilingual software.

----------♦----------

INTRODUCTION

Multilingual software development followed the various development approaches such as, structured programming approach, object oriented programming approach, component programming approach and architectural approach. Among the existing approaches, architecture based approach for multilingual software development is more advantageous. Therefore, architectural approach obtains its significance in the multilingual software development and it 1s the preferred approach by the developers like other domains.

Another advantage of using architectural approach is that it offers qualities like reusability, modifiability etc, From the multilingual perspective, the stakeholders demand qualities, that the multilingual software should exhibit The architect requires the qualities like understandability, reusability and language neutrality Developers demand maintainability quality. The end-user requires adaptability quality. Details about these qualities from the multilingual perspective are presented in the next section.

Multilingual software are designed at the detailed design step in the programming approaches Therefore, these approaches restrict the expected multilingual software qualities. In order to achieve these qualities, multilingual software have to be designed using architectural approach. But, the designers have the question that how to start the design and development of multilingual software using the architectural approach. This situation initiates the research questions to think about the software model for multilingual software development. Reference model for multilingual software also evolved over a period.

Different architectures of multilingual software are designed by applying ARMMS combining with different architectural styles like layered, client-server. interpreter. object-oriented, etc Architectural experience in building the multilingual architectures helped in fine tuning the reference model.

Initially, basic framework for multilingual software, which handles multilingual user interface, multilingual input-output. Multilingual and directory operations, is designed. Multilingual software like Kural, a multilingual programming language compiler and execution unit, Mayan, a multilingual web page design and dynamic connection tool, PONN Anjal. PONN SMS, a multilingual S MS utility, etc., are designed by applying A RMMS and using the basic multilingual framework. In this process, the basic framework evolved by capturing more and more multilingual requirements which mainly addresses language aspects.

Multilingualism got its significance in software due to globalization. In the present scenario, each human has to know more than one language for their official life and/or

social life. Therefore, the gadgets designed have to exhibit multilingualism. Human's key gadget, i.e. computer, also has to meet this requirement. In order to make computer multilingual, appropriate software has to be designed. Multilingual software is the one which exhibits multilingualism in one or more aspects like IO, UI, storage, process etc. The development of multilingual software is driven by its stakeholders namely software engineers, computational linguists, language engineers and end users. The stakeholders' requirements have been captured and realized in multilingual software. The multilingual software development efforts were made in software industries, academic institution and research and development organization.

In the software development process, in general, application domain expertise should be captured in order to document the key domain information in the form of a domain model and the model should capture the different views of the products from the perspectives of relevant stakeholders. Once the model is formed then it can be used as the vehicle for analysis and reasoning. Unfortunately, there is no reference model for multilingual software and also the multilingual software lacks in the non-functional qualities demanded by the stakeholders. This has made the multilingual software development more difficult.

In order to overcome the difficulties in developing multilingual software we have proposed an architectural reference model for multilingual software (ARMMS) by studying the existing multilingual software using unit operations. The layers of ARMMS are explained and the views of ARMMS are presented. A discussion on the qualities of the proposed model is also presented, which provides a basis for the future direction of multilingual software development process.

MULTILINGUAL APPLICATION DEVELOPMENT

A basic multilingual framework, named 'PONN', has been designed which addresses the multilingual services by the design of multilingual, language and language aspect components. The language aspects like character set, coding scheme and keyboard layout for the Tamil language, which are followed in the development of multilingual software is given as an example in the Appendix B. These components exhibit the functionalities in input-output, user interface and storage aspects. PONN has been developed as multilingual software for providing computer and its services in many languages especially in English and Tamil The basic multilingual framework has been improved based on the requirements of PONN and a multilingual standard application framework has been designed and developed This multilingual application framework has been used to develop the multilingual applications like SPECS, KURAL, PONN ANJAL and PONN Chat Initial development of PONN was done using VC++. Further, PONN has been ported in Java and this version has been tested with different operating systems like Linux and Solaris Parallel development of PONN has been earned out with more languages like Hindi and Malayalam. PONN has been also extended to work in intra and inter networks as N-PONN and I-PONN respectively.

Design of basic framework has been reused to design and develop the multilingual component based/ web application framework. Development of this framework has been carried out using Visual Basic and Component Programming in Visual Basic with ease due to the design reuse. This framework has been used to develop applications like MAYAN, a multilingual web page design and construction tool, multilingual VCD lending application and multilingual VISDATA (Multilingual Visual Database Manager).

In a similar fashion, multilingual mobile application framework has been developed and it is used for the application like, PONN SMS and Mobile Banking considering the limited resources like keypad, display etc These development efforts are hierarchically classified and shown in Figure 7.1. A brief outline about the multilingual software is presented below.

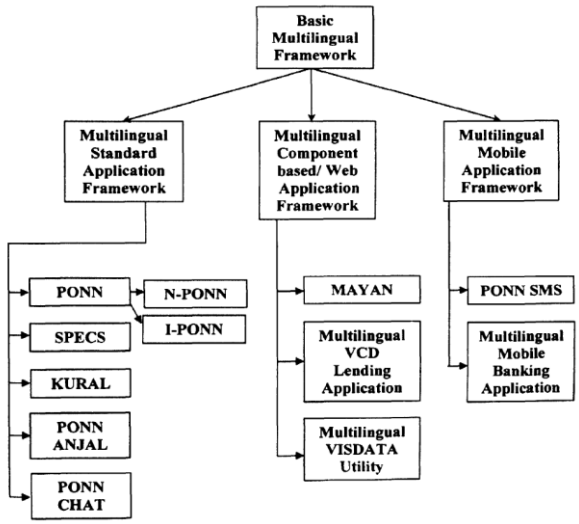

Figure 1: Development Hierarchy of Multilingual Software

Lots of research and commercial initiatives of developing multilingual software are going on. These initiatives are briefed below in order to gain a better understanding about the multilingual software development. According to Multilingualization group (m17n), multilingualization in

most software is peripheral, that is, multilingual facilities can be isolated from other (main) parts of the software. At the same time, most software has common requirement for their multilingual interfaces. A general library is being proposed that fulfils those requirements to make software development more efficient and inexpensive.

Domain level language requirements have a lot of significance in multilingual software development, we can find a framework for analyzing the multinational corporation (MNC) as a multilingual community in which parent functional language and subunit functional languages are concurrently used and recursively linked through an intra-corporate communication network.

These efforts inherently have some model but they are not explicitly revealed. Obviously we need an effort in this direction to extract the models of the existing multilingual software. Consequently these models help us in simplifying the multilingual software development process.

CHARACTERISTICS OF MULTILINGUAL APPLICATION

Today, the customers demand for quality software services. Among the various qualities, reusability is the primary quality which is demanded by the stakeholders of the software. Software architecture offers reuse in the various levels of software development life cycle (Garlan and Perry, 1995). Earlier the reuse is applied in the life cycle, greater the benefit that can be achieved. Reuse at the architectural level provides tremendous leverage for systems with similar requirements. For a given set of requirements which architecture is suitable can be stated by the designer out of his experience in building the reused architecture. When architectural decisions can be reused across the systems of different domains, all of the early decision consequences described by the designer are also transferred. On experiencing the multilingual software development, the requirements related to multilingual ism can be mapped towards an existing architecture Thereby reuse can be leveraged at the architectural level. A multilingual component can be modified frequently due to the non-standards of the digital form of languages. Hence, maintainability is one of the important nonfunctional qualities of multilingual software. So the qualities of multilingual software are essential Obviously, multilingual software should also offer non-functional qualities which are the common requirements demanded by the stakeholders of multilingual software The non-functional qualities of multilingual software namely maintainability, reusability, understandability, adaptability and language neutrality are derived from.

Maintainability : Maintainability, in the context of multilingual software, defines the extent of modifiability of a multilingual component after its deployment Modifications include corrections, improvements or adaptations to the multilingual components due to the changes in the language concerns. So, the maintainability of the multilingual component has direct attribute for measurement namely the customizability of multilingual concerns. If the customizability of a component is more, it leads to the easy maintenance of the component.

Reusability : Reusability, related to multilingualism, is the likelihood of an already developed/available segment of source code exhibiting multilingualism to be used again to add new multilingual functionalities with slight or no modification. Reusability of a multilingual component is the extent to which the component can be reused in the same and also in other multilingual applications.

Understandability : From the multilingual context, understandability is defined as a multilingual software quality which means ease of understanding multilingual software systems. In order to achieve this quality, the purpose of the multilingual software should be clear. The design and user documentation must also be clearly written so that it is easily understandable and the appropriate roles and responsibilities of language resource persons are assigned in the multilingual software development effectively. Understandability of design of multilingual software is essential to reserve the language resource persons, to increase the reuse and to reduce the frequent modifications in the multilingual software. Multilingual software is developed by a team of people namely, software developers, language engineers and end-users. Their understanding about the multilingual software characteristics and qualities is reflected in the clarity achieved in the design of multilingual software. Also, the existing approaches consider the multilingual characteristics of software only at the detailed design stage. This limits the understandability of the design of the multilingual software. But, the stakeholders expect the understandability of multilingual software which is essential for comprehensible multilingual software development.

Language Neutrality : In the current scenario, multilingual components are tightly coupled with domain components. Modification or introduction of multilingual components is more complex. Moreover, change in the domain component is also restricted due to this tight coupling. In order to overcome this, multilingual software should have language neutrality. This quality will enable the designer to develop language independent applications at domain level.

REFERENCE MODEL

It is clearly evident that software model is essential for better understanding, analyzing and designing software by the stakeholders of the software. Models enable the designers for making better planning and design which is vital in making the software. One can acquire more fine points about the software model. A reference model is a standard decomposition of a known problem into parts that cooperatively solve the problem. If the foundations of domain are firm and unlikely change radically, the model can be static. Otherwise, models will be changed and evolved. According to David L. Parnas, a model is a simplified or reduced size version of a product. Models have some, but not all, of the properties of the "real product" and are used because they are easier to study than the actual product. The relation of the model to the product i.e., which properties of the model are also properties of the product must be clearly stated. The views of stakeholders are addressed as, it is essential to have the repository of the organization's domain expertise to document the key domain information in the form of a domain model.

But these models have to be represented formally in order to understand and analyze them better. According to Mary Shaw and David Garlan which highlights the values of software design formalism, as it is generally agreed that formal models and techniques, are corner-stones of a mature engineering disciplines. Formalisms can be used to provide precise, abstract models and to provide analytical techniques based on these models. They are also useful for simulating behavior.

Reference model is a notion used in standard conceptual computing models. It is an abstract representation of the entities and relationships involved in a problem space, and form the conceptual basis for the development of more concrete models of the space, and ultimately implementations, in a computing context. It thereby serves as an abstract template for the development of more specific models in a given domain, and allows for comparison between complying models. Instances of reference models include, among others: the Open Systems Interconnection Basic Reference Model, the Open Geospatial Consortium reference models, the Von Neumann architecture as a sequential computing referential model, and the Federal Enterprise Architecture reference models.

A reference model is an abstract framework for understanding significant relationships among the entities of some environment, and for the development of consistent standards or specifications supporting that environment. A reference model is based on a small number of unifying concepts and may be used as a basis for education and explaining standards to a non-specialist. A reference model is not directly tied to any standards, technologies or other concrete implementation details, but it does seek to provide a common semantics that can be used unambiguously across and between different implementations.

MULTILINGUAL VCD LENDING SOFTWARE

The Video CD Lending application is designed to computerize the activities performed manually in a video CD lending library. This software enables the user to use the system in both Tamil and English. The video CD lending application provides facilities for the customer to select Video CD's according to his choice of actor, actress, category, language, director, music director and preview the film for making a better selection. It has an interface for returning the borrowed CD's. It generates a bill for returned CDs Facilities to add new CD details and artist details are provided It maintains CD, customer and transaction details for generating reports. All of the above functionalities work in the language of user's choice Visdata application is designed to provide facility for the user to create database, design tables, fields and indexes in multilingual data can be added, edited, updated and deleted to and from the table.

MAYAN, A Framework for Multilingual Web Page Creation and Dynamic Construction, developed as a multilingual framework using which a website designer can construct web pages with the same page structure and content in different languages. Using this framework, a designer can store all the versions of a web page together, so that they all have to be updated easily when required It provides an efficient content management, separating the content from structure.

PONN SMS is developed to provide the Short Message Service (SMS) in mobile phones in multilingual form PONN SMS provides a multilingual environment, which facilitates the user to communicate with mobile phone/computer in multilingual form This multilingual interface has to change according to the user's choice of language.

Mobile banking is the software that is designed to allow all types of banking transactions to be carried out over mobile devices All banking operations, which does not need the presence of customer such as ordering a cheque book, balance enquiry, loan eligibility check etc, are done using this software in the user's preferred language.

RESEARCH APPROACH

In order to carry out the research work in multilingual software development and to prove the research hypotheses, formal representation of the models,

technique for analyzing the model and operations to form the model are required.

The designers' knowledge, which IS expressed In the multilingual software, is to be captured and formed as domain models More details about the software model can be obtained from (Bass et al., 2002, Parnas, 2001 ; Clements and Northrop, 2002). Models of multilingual software have to be mined from the existing multilingual software. But, model has to be represented formally In order to better understand and analyze According to (Shaw and Garlan, 2000), formal models and techniques are essential for formal analyses. Formalisms can be used to provide precise, abstract models and these models are useful for simulating behavior.

Formalizing the models can be achieved using Z language (Sptvey, 1989) or CHAM model (Inverardi and Wolf, 1995). Formal~zation IS also carried out with UML (Clements et al., 2004; Fowler and Scott, 2000) even though it has some limitations. The mathematical structures (Cohoon et al., 2006; Henno, 2006) are significant way of carrying out formalization. It is important to decide the formalization technique to formalize multilingual software models with simplicity.

Design of multilingual software is complex due to the non availability of models or methodologies for it. The complexity IS more due to multilingual software quality requirements and their overriding characteristics. In order to address this, the multilingual software requirements have to be analyzed with the help of design space approach to overcome the complexion in design process. A key part of the design space approach is to choose the dimensions that reflect requirements and structure (Lane, 1990). This reflects the correlation found among such dimensions that provides direct design guidance This design guidance will be helpful in proposing a new model.

Reference model can be evolved from the concrete model by applying unit operations (Kazman et al, 1993) The architectural experience of developing multilingual software by applying the reference model can be reported. It will help the multilingual software designers to reuse the existing model/architecture/components and also it will aid the designer to further improve this model for multilingual software development In future

CONCLUSION

A study of literature was carried out to understand the multilingual software and its characteristics. The classification of multilingual software and the classification of multilingual software development techniques were also presented in order to obtain a clear understanding about the evolution of multilingual software.

Design space approach was used to analyze the models in order to overcome the inadequacies. The requirements of multilingual software have been analyzed using the design space approach and the factors which limit the qualities of multilingual software were identified An aspect based language library model was proposed and it was formally represented using the algebraic structure. Unit operations were used to refine the aspect based language library model into reference model Different multilingual software architectures were designed by applying ARMMS combining with different architectural styles like layered, client-server, interpreter, object oriented etc, using object oriented approach and UML Architectural experience in budding the multilingual architectures helped In fine tuning the model.

Multilingual software qualities like maintainability, reusability, understandability, adaptability and language neutrality are presented in this paper which is derived from the existing approaches. The research questions and hypotheses about the multilingual software development are stated. The research approach is also briefed.

Change is the on14 permanent process in the world Architecture reference model for multilingual software should get changes In terms of evolution and improvements. The future work concerned with ARMMS can be stated with respect to two perspectives, namely language perspective and architectural perspective. From the language perspective, more languages can be made suitable for ICT applications and used in the PONN framework.

REFERENCES

- Multilingual terminal: a universal approach, J.Dougnac, Télécom Review, Paris, France, Dec 1995
- Measuring the Performance and Intelligence of Systems: Proceedings of the 2002, Edited by: E.R. Messina and A.M. Meystel, PerMIS Workshop, 2002
- Multilingual Technology and Tools for IT Developments in India, S. Kuppuswami, T.Chitralekha, and V. Prasanna Venkatesan, Technorama, India, October 2003
- Software Architecture in Practice, Len Bass, Paul Clements, Rick Kazman, Pearson Education, Third Indian Reprint, 2002
- Software Architecture: Perspectives on an engineering discipline, Mary Shaw and David Garlan,

Prentice-Hall of India private limited, 2000

- Software Fundamentals: Collected papers by David L. Parnas edited by Daniel M. Hoffman and David M.Weiss, Pearson Education, 2001

- Software Product Lines: Practices and Patterns, Paul Clements and Linda Northrop, Addison Wesley, 2002

- The multinational corporation as a multilingual community: Language and organization in a global context, Yadong Luo, Oded Shenkar , Journal of International Business Studies, Vol 37, 2006

- Toward a Software Engineering Model of Human-Computer Interaction, R. Kazman, L. Bass, R. Little, Engineering for Human-Computer Interaction, Proceedings of the IFIP WG2.7 Working Conference, North Holland, August 1993, 131-153.