Chapter 3

DESIGN OF MULTILINGUAL SOFTWARE USING ARCHITECTURAL APPROACH

3.1 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 is 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. These research questions help in forming the research hypothesis and they are presented in this chapter.
3.2 Qualities of Multilingual Software

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 multilingualism 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 (Acharya, 2007, Schmitt, 2001, M17N, 2007, Indiainteractive, 2007).

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.
In the case of component maintainability, modification of language component is recurrent due to the frequent change requests in the language concerns which are initiated from the stakeholders. Maintaining the multilingual component library is also tedious due to frequent updates in the existing language components and also addition of new languages. Easy maintainability of the multilingual components is the requirement of the stakeholders.

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.

Multilingual software requires reusability of multilingual components in their software development which will save the development cost and time. But this quality is limited due to two factors. First one is the different standards of a language followed by the development community in different geographical locations. These multiple standards create software silos. Secondly, the frequent modifications of multilingual components are due to the non standard digital forms of the languages. Both of these limitations have to be addressed to achieve the reusability. Reusability is the major concern of the stakeholders of multilingual software.

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.

**Adaptability**

Multilingual software should be able to work with multiple languages concurrently based on the users’ choice. Selection of the working language, which is a multilingual software characteristic, is carried out statically or dynamically. In the static approach, the user will configure by passing parameters or configuration files. But in the dynamic approach the languages are selected at run time whenever the user needs the change. Multilingual software should be able to bind the language resources at compile time or run time to enable it to work in the required language. Language adaptability is one of the qualities demanded by the stakeholders of multilingual software.

**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.

These multilingual software qualities have to be considered in the multilingual software design using the architectural approach.
3.3 Problem Statement

Multilingual software has to be designed to suit the requirements of stakeholders. These requirements can be mapped into multilingual software characteristics and/or multilingual software qualities. So, complete multilingual software is the one which exhibits multilingualism in all aspects like I/O, user interface, process etc, and having the required multilingual software characteristics and multilingual software qualities.

To develop complete multilingual software, an architectural approach is the highly suitable approach. In order to apply the architectural approach, no explicit models or methods are available specific to multilingual software. So, designing the model of software is the first step towards architecture in the design process. When the model is combined with an architectural style, it will result in an architecture. Model of compiler or networking software is the best example for the use of models in the design process. So, this situation initiates the following set of questions which are relevant for application of architectural approach for multilingual software.

- Are there any models for multilingual software which will simplify the multilingual software development?

In order to start the design process, the models for multilingual software are essential. But there are no explicit models available for multilingual software development. Existing multilingual software development approaches inherently have some models and they are not reported explicitly. Mining has to be carried out on the existing multilingual software for the models. These models have to be analyzed for the expected multilingual software qualities. Based on this the following research hypothesis is stated.

Hypothesis I

The design rationale behind the existing multilingual software has to be mined and formally modeled which consists of the multilingual concerns. An analysis of the mined models has to be carried out based on the multilingual software qualities.
• If models exist, whether they provide the expected multilingual software qualities?

The existing multilingual software are designed using their own implicit models. The designer has to analyze, what are the inadequacies in the mined multilingual software models. These inadequacies have to be addressed by proposing a model for multilingual software. This forms the second hypothesis and it is stated below.

**Hypothesis II**

**Based on the analysis of mined models, separation of the language aspects has to be carried out and it has to be used to form a new model for multilingual software development. This proposed model has to be refined into a reference model which exhibits the expected quality requirements.**

**3.4 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 analysis. 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 (Spivey, 1989) or CHAM model (Inverardi and Wolf, 1995). Formalization 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 complexity 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.

3.5 Summary

Multilingual software qualities like maintainability, reusability, understandability, adaptability and language neutrality are presented in this chapter 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.