Chapter 7

Conclusions and Future Work

The quantification of software quality factors is essential for analyzing the quality of software. Both researchers and software developers have felt the need for developing the metrics for quantification of various attributes of a software system. Identifying attributes that affect a specific quality factor is first step towards quantification. Designing the metrics and assessing the applicability of metrics in predicting/quantification of quality factors is equally important. Empirical studies have been found to be effective in validating the applicability of metrics. In this thesis we have presented an empirical study for validating the applicability of CK metrics, using open source software systems, in predicting fault proneness of class. It was observed that some of the metrics like RFC, WMC were important for predicting the fault proneness of a class. We also conducted study to find out the effect of software revisions on metrics value. The applicability of Lehman’s laws of software evolution to open source software was studied in reference to CK metrics. We observed that the reflection of some of the laws namely law 1, law 2, and law 6 were easily determined using the metrics. But the relatedness of law 3, law 4 and law 5 to open source software systems was difficult to determine.

Complexity of a software artifact have been extensively studied by the researchers. One of the factor affecting the complexity of software is crosscutting requirements. The thesis reports on the approaches and tools available to identify the crosscutting requirements. This will help the developers to identify crosscutting requirements early in the software development process. The knowledge of existence of such concerns in early
developmental phases can help in handling them more efficiently.

Several researchers have proposed metrics to quantify quality factors, quality criteria or software attributes. There are still many quality factors and criteria for which metrics does not exist. One such criteria is software code understandability. Understandability of code is extremely important for maintaining it. In this thesis we have reported the metrics designed to measure the cognitive complexity of a class, due to its attributes, integral to Object Oriented Software System (O OSS). The measure of cognitive complexity can be used for computing the maintainability of a class.

The quality factors such as accessibility has received little attention from researchers community. Specially the accessibility of web based applications on hand-held devices. Metrics are required to measure the accessibility of web applications on devices with limited capacity. This thesis reports the metrics to measure the accessibility of web based applications on hand-held devices.

Interoperability is another quality factor which has gained attention of researcher community. Software developers face challenges in making the applications interoperable specially when the software developed using different frameworks/ platforms etc. Although there are tools to handle interoperability issues and some of the tools are reported in this thesis also but still the interoperability between software modules/ applications is not seamless. Interoperability problem deepens when the software applications have to interoperate over Internet. This thesis reports some issues in interoperability of web services where there are challenges in information exchange between a web service an its clients.

The research work reported in this thesis will help the software development community in following ways:

1. The results of empirical study for validation of various object oriented metrics reported in this thesis will further enhance the confidence of software developers in these metrics.

2. The tools and approaches listed in this work may help the developers in identification of cross cutting requirements and subsequently, a mechanism may be adopted to handle such requirements.
3. Using the metrics to measure the cognitive complexity of class being developed, software developers may choose to redesign a class in case its cognitive complexity is high.

4. The accessibility metrics reported in the thesis will prove useful to developers, designing web applications, in evaluating the accessibility of the web applications.

5. The interoperability issues reported in this work can be used by the software community in resolving the interoperability problems in applications.

7.1 Future Work

The work can further be extended by doing more empirical validations of applicability of cognitive complexity metrics in predicting understandability. The maintainability of software module, apart from understandability, depends upon changeability also. The metrics to measure the changeability of a software module can be defined and may further be validated for its applicability in measuring changeability. The measure of changeability can further be extended to measure maintainability of software systems/modules.

The existence of crosscutting requirement has been shown to cause defects. The effect of crosscutting requirement on other quality factors and quality criteria can also be studied. It is very likely that software modules dealing with crosscutting requirement are difficult to change and subsequently difficult to maintain.

The attributes affecting interoperability of a software application can be computed for an open source applications and using an empirical analysis the effect of attributes on interoperability can be established. Further, some metrics for interoperability can be designed using the validated attributes. All the existing metrics can be further validated for their applicability in predicting the various quality factors.