The aim of this chapter is to draw conclusions of work so far done and presented in this thesis. The chapter also addresses some potential aspects of the software defect density prediction that need enhancement in future.

8.1 CONCLUSIONS

The works proposed through this thesis; software developers are able to identify the possible error prone module/class of the software product under development. This prediction used by the developers to redistribute the resource and pay more care on the highlighted parts in the software products by this approach. The approach has been validated by both methods of development, i.e. commercial development and open source development. As highlighted by several studies, the nature of the defects is random distribution amongst the modules, hence, this nature force software developer to predict the latent defects in developing a software product. Otherwise, the developed product does not attain the acceptable level of quality; lack in meeting specification, customer satisfactions, delivery schedule and other quality criteria.

Software defects are considered to be a prime factor that affects the overall quality of the product. Most of the studies used the different prediction techniques to predict the number of latent defects in the product using some historical information/data of the product. These studies overlooked the size factor that is needed to be analyzing with combining to a number of defects. Software defect density measure can be considered as a factor that combined represents the number of known defects with the size of the module/class in a software product.
The work done in thesis has successfully accomplished the following objectives:

I. A model has been proposed that established the linear relationship between software defect density and module size. The analysis has been performed on three different data sets and concluded that the module size distribution on geometric fashion leads to minimize the defect density in the developing product. The model empirically validated using the goodness of fit test on the studied data sets.

II. A relationship between size and defect insertion rate has been investigated and validated by past studies. Although the proposed model for defect density prediction is able to minimize the defect density but the parameters are used in additive nature limited to a few values. An effective model has been proposed that established the relationship between size and the defect density using multiplicative factors. This model also minimizes the defect density.

III. Prediction of faulty or non-faulty modules/classes using classification techniques does not priorities the modules in the faulty class. Prediction of defect density in modules/classes motivates the developer to prioritize the module on the basis of defect density. An approach has been empirically validated for prediction of defect density using static code metrics.

IV. In the traditional static metric suites, some of the metrics are not showing significance relationship with the defect density. Hence, a hybrid metric suite has been formulated using machine learning approach, and clusters have been formed of similar metrics from this suite. The result shows that most of the insignificance metrics from the traditional metrics suites showing a significant relationship with defect density.

V. Prediction of defect density in open source software using static code metrics suite requires lots off effort to extract the information for these metrics from the CVS
database. An easiest and simple repository metrics suite has been used to predict the defect density in Open Source Software.

8.2 SCOPE OF FUTURE WORK

Future work in the following direction is suggested:

- In the model, proposed in Chapter 4, only two parameters are added that limits the dependency of defect density only on these parameters. Adding more parameters to the model enhance the performance in terms of defect density minimization.

- In chapter 5, the publically available data set has been used for empirical validation of defect density prediction using some static code metrics. The approach can validate on the real data set from the industries, and more metrics can be added in the prediction model for enhancing the prediction performance.

- In Chapter 7, few repository metrics didn’t show the significance relationship with defect density. New metrics can be added in the metrics suite to investigate the significance relationship with defect density.