Chapter 7

Conclusion

The overall objective of this thesis is to prescribe certain preprocessing techniques that facilitate requirement based software testing. The aim is to improve the efficacy of test scenario generation, prioritization and selection. Test scenario management using an ontology is also discussed. This chapter, presents a comprehensive summary of the work carried out in this thesis. A proposition on areas for future work concludes the chapter.

7.0.1 Summary

Given the size of software systems, there is need to automate the process of testing. Further, for reduction of test effort, certain preprocessing like test scenario generation, prioritization and selection is to be carried out. The Unified Modeling Language (UML) is being increasingly used in the industry to gather requirements and design of a system. Testing using specification (here, UML) is an important topic of research due to the advantage involved in using specification captured during the requirements gathering and analysis phase for testing. It helps by aiding different stakeholders in understanding different aspects of the system as well as helps to reduce defects in design and implementation as the same UML diagrams
can be used across the software development life cycle.

This work has contributed to research in the following aspects. First, a method for inter-model consistency checking among the use case, activity, sequence, class and state diagram is proposed. Inconsistency creeps in among the UML models as different associates are involved in different phases of the SDLC and bring in their own perspectives. These are to be viewed together to ensure that the inconsistencies do not propagate to design errors. Consistency is ensured by framing well-formedness rules and applying these on the UML diagrams that specify a software.

Second, a method for scenario generation is introduced, specifically in case of concurrent activities. The different scenarios of an application are to be analyzed in order to identify test scenarios. The order in execution of activities is represented in activity diagrams. Concurrent activities are represented using fork-join constructs. The number of scenarios to be tested in presence of a simple fork-join construct could be large. This work exploits the concept of dependency that exists between concurrent activities to aid in reducing the number the scenarios generated. Dependency is used by assigning priorities and levels at which activities are executed. The proposed method helps avoid repeating scenarios thereby reducing the cost and effort involved in testing.

Third, a method for prioritizing scenarios is introduced. The objective is to order scenarios in a manner that aims at detecting defects early and maximizing coverage. Prioritization is done by taking a weighted sum of both customer inputs as well as structural aspects of the software. Customer priority is obtained by involving the customer(providing a prioritization of the features as per the expected usage requirements). Structural priority is calculated by considering the primitives of both use case and activity diagrams. Weights are given to the constructs and priority calculated. The priority of a use case is due to combined
priority of customer inputs and structural priority. Scenarios are prioritized based on structural complexity only.

Given constraints of cost and time, it is difficult to test all scenarios. Hence, test scenario selection, aims at selecting a subset of scenarios for testing based on similarity that exists among scenarios. Three techniques, based on similarity measures have been proposed. In the first, Levenshtein distance is used to calculate the similarity between scenarios and used to obtain a subset of scenarios. The second method takes the weight, length and position of the subscenarios into consideration to calculate similarity. In this work, it is presumed that two scenarios are more similar if the relative position of the subsceanrios in the two scenarios is the same. The third method for selection is based on clustering. Clustering provides the advantage of grouping similar scenarios for selection.

An approach for test scenario management is proposed. The increasing size of software systems makes it necessary to manage the test scenarios in an efficient way. In this regard, ontologies help to analyze the relationship between requirements captured using UML diagrams. Scenarios generated from corresponding activity diagrams along with activities, are linked to use cases. Further, software testing concepts adopted from SWEBOK, are used to build an ontology for test management.

Tools implementing the proposed methods are developed and case studies are carried out. The results obtained due to case studies are encouraging: First, the efficacy of rule based consistency checking is demonstrated. The approach followed for scenario generation shows that the use of dependency between activities is an effective way to eliminate unwanted scenarios, particularly in case of concurrent activities. Two ways of doing this(i.e. priority and level of activities) has been proposed, which are effective in reducing the number of scenarios.

The results obtained for prioritization show that automated prioritization tech-
niques to order use case scenarios is effective and can be used in combination with customer assigned priority. The technique achieves a high rate of fault detection. Similarly, the techniques used for test scenario selection show that distance measures can be effectively used as a basis for testing a subset of the entire set of scenarios. Finally, the result obtained in the case studies demonstrate and validate use of the proposed prioritization and selection techniques in testing.

With regards to the ontology for test scenario management, the benefits it provides are two fold. First, it provides a way to infer and reason on use cases and scenarios for testing. This helps in finding undefined relations between use cases thereby helping in test management. Secondly, it provides the users a common language to communicate and query to understand test needs.

7.0.2 Future Work

There could be several natural extensions to this work. In this work, the focus has been on functional requirements only for prioritization of scenarios. An obvious extension could be prioritization considering other functional and non-functional requirements of a system. Also, it is important to consider risk factors while providing weights to activities for prioritization. A new metric to quantify test coverage following the proposed measure needs to be investigated. Also, there is need to study different measures to use for clustering based selection and analyze its suitability. Further, it will be interesting to investigate usage of ontology for all the steps like scenario generation, prioritization and selection dealt here.

7.0.3 Publications

Parts of the present work has evolved into the following conference and journal papers:

1. 'Ensuring Consistency in Relational Repository of UML Models',


7. 'Clustering Test Cases to Achieve Effective Test Selection’, Amrita ACM-W Celebration of Women in Computing, First Conference of Women in Computing in India. (accepted)


9. 'Developing an Ontology for Test Scenario Management’, 7th International Conference on Distributed Computing and Internet Technology. (communicated)