

Preface

In this book we introduce SpecDB, a database created to represent and host software specifications in a machine-readable format. The specifications represented in SpecDB are for the purpose of unit testing database operations. A structured representation aids in the processes of both automated software testing and software code generation, based on the actual software specifications. We describe the design of SpecDB, the underlying database that can hold the specifications required for unit testing database operations.

Specifications can be fed directly into SpecDB, or, if available, the formal specifications can be translated to the SpecDB representation. An algorithm that translates formal specifications to the SpecDB representation is described. The Z formal specification language has been chosen as an example for the translation algorithm. The outcome of the translation algorithm is a set of machine-readable formal specifications.

To demonstrate the use of SpecDB, two automated tools are presented. The first automatically generates database constraints from represented business rules in SpecDB. This constraint generator gives the advantage of enforcing some business rules at the database level for better data quality. The second automated application of SpecDB is a reverse engineering tool that logs the actual execution of the program from the code. By Automatically comparing the output of this tool to the specifications in SpecDB, errors of commission are highlighted that might otherwise not be identified. Some errors of commission including coding unspecified behavior together with correct coding of the specifications cannot be discovered through black box testing techniques, since these techniques cannot observe what other modifications or outputs have happened in the background. For example, black box, functional testing techniques cannot identify an error if the soft-

ware being tested produced the correct specified output but more over, sent classified data to insecure locations. Accordingly, the decision of whether a software application passed a test depends on whether it coded all the specifications and only the specifications for that unit. Automated tools, using the reverse engineering application introduced in this book, can thus automatically make the decision whether the software passed a test or not based on the provided specifications.

This book is intended for to the members of the software testing team and developers of software testing tools. System analysts and designers could also benefit from the ideas presented in this book. Although the title of the book caters for database applications, yet, those interested in software applications that do not involve database systems can also benefit greatly from the concepts in this book, since applications based on database systems are a superset of those that do not access databases.