Preface

Testing software is a very important and challenging activity. This is a book for people who test software during its development. Our focus is on object-oriented and component-based software, but you can apply many of the techniques discussed in this book regardless of the development paradigm. We assume our reader is familiar with testing procedural software—that is, software written in the procedural paradigm using languages such as C, Ada, Fortran, or COBOL. We also assume our reader is familiar and somewhat experienced in developing software using object-oriented and component-based technologies. Our focus is on describing *what* to test in object-oriented development efforts as well as on describing techniques for *how* to test object-oriented software, and how testing software built with these newer technologies differs from testing procedural software.

What is software testing? To us, testing is the evaluation of the work products created during a software development effort. This is more general than just checking part or all of a software system to see if it meets its specifications. Testing software is a difficult process, in general, and sufficient resources are seldom available for testing. From our standpoint, testing is done throughout a development effort and is not just an activity tacked on at the end of a development phase to see how well the developers did. We see testing as part of the process that puts quality into a software system. As a result, we address the testing of all development products (models) even before any code is written.

We do not necessarily believe that you will apply everything we describe in this book. There are seldom enough resources available to a development effort to do all the levels and kinds of testing we would like. We hope you will find a number of approaches and techniques that will prove useful to and affordable for your project.

In this book we describe a set of testing techniques. All of the techniques we describe have been applied in practice. Many of these techniques have been used in a wide variety of industries and on projects of vastly different sizes. In Chapter 3, we will consider the impact of some of these variables on the types of testing that are routinely performed.

To describe these techniques, we rely in many cases on one or more examples to illustrate their application. We hope from these examples and from our explanations that you can apply the same techniques to your project software in a

straightforward manner. The complete code for these examples, test code, and other resources can be obtained from http://cseng.aw.com/book/0.3828.0201325640.00.html.

In order to make this book as useful as possible, we will provide two major organizational threads. The physical layout of the book will follow the usual sequence of events as they happen on a project. Model testing will be addressed earlier than component or code testing, for example. We will also include a set of questions that a tester might ask when he or she is faced with specific testing tasks on a project. This testing FAQ will be tied into the main body of the text with citations.

We have included alternative techniques and ways of adapting techniques for varying the amount of testing. Testing life-critical or mission-critical software requires more effort than testing an arcade game. The summary sections of each chapter should make these choices clear.

This book is the result of many years of research, teaching, and consulting both in the university and in companies. We would like to thank the sponsors of our research, including COMSOFT, IBM, and AT&T for their support of our academic research. Thanks to the students who assisted in the research and those who sat through many hours of class and provided valuable feedback on early versions of the text. The consultants working for Korson-McGregor, formerly Software Architects, made many suggestions and worked with early versions of the techniques while still satisfying client needs. The employees of numerous consulting clients helped us perfect the techniques by providing real problems to be solved and valuable feedback. A special thanks to Melissa L. Russ (formerly Major) who helped teach several tutorials and made her usual insightful comments to improve the material.

Most of all, we wish to thank our families for enduring our mental and physical absences and for the necessary time to produce this work: Gayle and Mary Frances McGregor; Susan, Aaron, Perry, and Nolan Sykes.

JDM

DAS