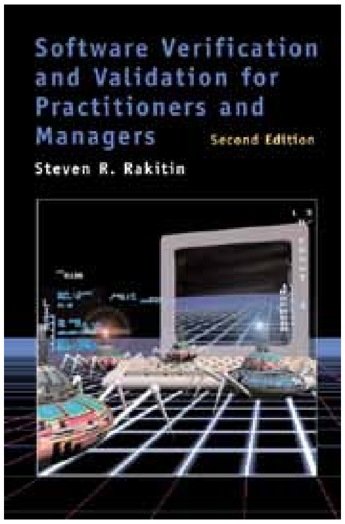
Software Verification and Validation for Practitioners and Managers

Second Edition, by Steven R. Rakitin

Preface


Software Verification and Validation for Practitioners and Managers
Second Edition
Steven R. Rakitin

The book contains what I believe to be a reasonable set of basic Software Verification & Validation (V&V) activities. From firsthand experience working with many companies in several industries, I've found that basic software V&V activities are not well understood and are applied inconsistently. For example, several companies I've worked with recently still did not have a Software Quality Assurance (QA) group and a few hadn't even implemented basic Configuration Management (CM) practices. Further, in those organizations that had Software QA, I've found wide variances in effectiveness across organizations and even across projects within the same organization. While my observations are not based on a statistically significant sample, I believe that my experiences are a fairly accurate reflection of the industry as a whole. In fact, Yourdon reported recently that, while software quality has improved overall, the gap between those companies' producing the best and worst software has increased dramatically over the past decade.

As a consultant, I'm frequently asked to help companies that are having "quality problems". Studying clients that were having "quality problems", I found a common theme – they all behaved in an *unpredictable* manner. For example, it wasn't possible for these organizations to determine when major events, like code freeze or first customer ship, would happen. (In some cases, it was not even possible to know if they would happen.) Because these organizations were unpredictable, Marketing was unable to plan product rollout events, Development and QA were unable to make effective use of expensive, *scarce resources and, software V&V activities were not nearly as effective as they could be.*

Len Race, a consultant and friend, observed that in order for businesses to become more efficient, they must learn how to behave in a more predictable manner. For me, this was a revelation. The connection between poor performance and unpredictable behavior became crystal clear. Unpredictable organizations exhibit many of the following symptoms:

- They consistently commit to more than they can deliver and consistently deliver less than was committed.
- They underestimate tasks and miss most every schedule.
- People within the organization have goals and objectives that are not aligned with the overall business goals and objectives.
- There's a lack of accountability throughout the organization.
- There is no notion of using "Best Practices". If a written process exists, it's not followed consistently.
- There is a belief among employees that "We never have time to do things right but always have time to do things over".

When I help a company with "quality problems", I start with the CEO and ask "How are your people measured?" By looking at performance plans for individuals, you can understand why the organization behaves the way it does. Without fail, when looking at performance plans in companies with "quality problems", I rarely come across the word "quality". *Since behavior is directly related to how people are measured, why is it a surprise that these organizations have "quality problems"?*

Once I made the connection between poor performance and unpredictable behavior, I realized that:

- *Management owns both the problem and the solution and*
- *To increase the effectiveness of the software V&V techniques described in this book, organizations have to learn to behave in a more predictable manner.*

Clearly, Management must take a leadership role in helping their organization behave in a more predictable way. It is for this reason that the title of this book has been changed to Software Verification and Validation for Practitioners and Managers. Specific actions that Management should take that will help their organization learn to behave in a more predictable manner have been included in this second edition.

Who is this book for?

This book is intended for two groups – practitioners and managers.

- Practitioners include software QA engineers, software engineers, and project managers who need a basic understanding of Software V&V techniques.

  Unfortunately, very little in the way of formal training in software V&V is offered in school. As a result, there is a gap between the skills that many software quality practitioners have and the skills that are needed to produce high quality software. This book is intended for those people who have been given responsibility for performing software V&V tasks but who have not had the luxury of receiving any training in this subject. Parts I-III are intended primarily for practitioners.

- Managers include Software QA Managers, Development Managers, Project Managers, Vice Presidents of Engineering/Development, Directors of Quality, and CEOs.

  Management has the ability to change the behavior of the organization. How, by providing the leadership necessary to meet business goals and by aligning the way people are measured with those same business goals. Every manager from the CEO to first line managers need to recognize that meeting business goals will be easier when the organization becomes more predictable. Part IV of this book was written to provide managers and executives at all levels with specific strategies they can use to help their organization behave in a more predictable manner.

For what kinds of software should software V&V activities be used?

The software V&V activities described in this book are applicable to a wide range of software, a wide variety of products, and a wide range of industries. The best way to answer this question is with another question: "Is there a compelling business reason to develop good quality software and deliver it on time?" If the answer is "Yes", then many of the activities are applicable.

How is this book organized and what's changed in this edition?

- Part I provides an introduction to software development and an overview of the software development process. Several software development life cycle models are presented in Chapter 2. The importance of a written Software Development Process is outlined in Chapter 3. Economic motivation for many software V&V activities is discussed in Chapter 4.

  In Chapter 1, a discussion of the international standard on Life Cycle Models ISO 12207 has been added. In Chapter 2, information on the Rational Unified Process has been included. References to software standards have been updated throughout Part I.

- Part II provides an overview of Software Verification activities. Chapter 5 and 6 (along with Appendices A-D) provide details on the Formal Inspection Process. Chapter 7 focuses on verification measures and Chapter 8 contains an overview of Configuration Management.

  Chapter 6 has been reorganized slightly to remove some redundant information. References to software standards have been updated throughout Part II.

- Part III provides and overview of Software Validation activities. Chapter 9 provides an overview of Testing. Testing measures are included in Chapter 10 and an introduction to Software Reliability Growth is presented in Chapter 11.

  Chapter 9 has been rewritten to include more details about types of tests that can be written and now includes information on the Concurrent Testing/Development Model, test planning, and test estimation techniques. Chapter 10 has been reworked to focus more specifically on validation measures.

- Part IV is new and is focused on providing Management with specific strategies they can use to help their organization learn to behave in a more predictable manner, thus significantly improving the effectiveness of software V&V activities. Chapter 12 provides an introduction and economic motivation. The topic of balancing Quality, Features, and Schedule is discussed in Chapter 13. Chapter 14 presents the Yellow Sticky Method - an accurate method for estimating tasks and building realistic schedules. Chapter 15 discusses issues related to balancing the needs of People, Process, and Product. Techniques for managing commitment and risk are discussed in Chapter 16.

  In addition to Part IV, 3 new appendices have also been added.

Acknowledgements
The notion of Predictable Software Development evolved from discussions I've had with my friend and colleague Len Race. His insight into business processes and his ability to "get things done" is simply uncanny.

I would also like to acknowledge the staff at Artech House. I am indebted to Tim Pitts, Ruth Young, Judi Stone, and others unbeknownst to me for helping me through the publishing process. They have provided a constant source of patience and encouragement.

Steven R. Rakitin
Upton, Massachusetts
July 2001