**Testing Dirty Systems**

“Traditional testing approaches generally do not work with dirty systems. A different approach is needed if testing is going to be effective. However, the tailored approaches need merely to be extensions of good testing practices.

“In order to understand the difference between common testing approaches and the approach needed for dirty system testing, let’s look at a coffee-making analogy. Let’s assume that we have brewed a pot of coffee and it is filled with specks of coffee bean. These bean specks are errors or defects in the coffee. The objective of testing is to remove these defects. Traditional approaches might involve the use of tweezers to pick out the bean specks one at a time. Although after enough time and effort all of the bean specks would be removed, by then the coffee would probably be cold and undesirable. The dirty system approach proposed in *Testing Dirty Systems* would be to pour the coffee through a filter. The filter would catch all of the bean specks and we would have hot, defect-free coffee in almost an instant.

“Dirty system testing incorporates the experiences of leading test organizations over long periods of time. It incorporates good testing practices together with knowledge of why systems fail. In essence, the approach incorporates the combined experiences of many testers into an effective test practice.”

—from Chapter 2

**About the Authors**

William E. Perry is the executive director of the Quality Assurance Institute (QAI) based in Orlando and author of more than fifty books on quality assurance in data processing. He served on the 1988 and 1989 board of examiners of the Malcolm Baldrige National Quality Award.

Randall W. Rice is a consultant and instructor on software and systems testing. Based in Oklahoma City, he is the principal consultant of Rice Consulting Services and has more than 25 years’ experience in building and testing information systems. He is also publisher and editor of *The Software Quality Advisor* newsletter.

---

**Partial Contents**

- **The Dirty System Challenge**
  - What is a Dirty System?
  - How to Measure the Dirtiness of the Software to Be Tested
- **The Six-Step Process for Testing Dirty Systems**
  - Dirty Systems Can Be Tested
  - The Importance of Having a Process for Testing Dirty Systems
  - The Process for Testing Dirty Systems
- **Step 1: Dirty System Diagnostic**
  - Step
  - The Black Holes of System Knowledge
- **Step 2: Dirty System Test Planning**
  - The Impact of Dirty Systems on Test Planning
  - How Tools Can Help in Designing Test Cases
  - Types of Test Cases
  - The Role of Regression Testing
  - Test Results
  - Test Scripts
- **Step 3: Dirty System Test Execution**
  - The Twenty Most Common Software Problems
  - How to Test Each Problem Area
  - Requirements Management Tools
  - Managing Change in Testing Dirty Systems
  - Change Management Process
  - Dirty System Test Execution by Test Phases
  - Dirty System Test Execution by Test Types
  - Manual versus Automated Testing of Dirty Systems
- **Step 4: Dirty System Test Results Analysis**
  - Impact of System Dirtiness on Test Analysis
- **Step 5: Writing Dirty System Test Reports**
  - Getting the Most Value from Defect Reports
  - Defect Tracking Tools
- **Step 6: Using Test Results to Clean Up the Dirty System**
  - The Difference Between Quality of Design and Quality of Performance

**From the Authors of Surviving the Top Ten Challenges of Software Testing, Guidance on Testing the Toughest Systems**

Some systems are more difficult to test than others. Software testers contend with undefined or partially defined requirements; outdated, incomplete, or nonexistent documentation; complex logic; a mixture of languages; or worse. All of these factors make a system dirty, or virtually untestable.

In *Testing Dirty Systems*, authors William Perry and Randall Rice teach testers a six-step process for approaching such systems: system diagnosis • test planning • test execution • test analysis • report development • dirty system repair.

Because of the unknown characteristics of the dirty system, the traditional validation of comparing actual processing results against the expected processing results is often inadequate. Analysis of a dirty system must go much further into describing the expected operational characteristics of the system, including

- probability of failure based on failures during testing
- expected difficulty of making changes based on inadequacy of documentation
- estimate of defects that remain in the system
- operating conditions that will lead to failures
- coverage levels based on code or test cases
- complexity levels based on coding structure

Project leaders, independent testers, quality assurance personnel, and IS auditors will benefit from this book, as well as end-users and customers with a vested interest in the success of their systems.

Read more about this book at [www.dorsethouse.com/books/tds.html](http://www.dorsethouse.com/books/tds.html)