1. Record Nr. UNINA9911006623403321

Autore	Zeller Andreas
Titolo	Why programs fail : a guide to systematic debugging / / Andreas Zeller
Pubbl/distr/stampa	Burlington, MA, : Morgan Kaufmann Oxford, : Elsevier Science [distributor], 2009
ISBN	1-282-16881-9 9786612168819 0-08-092300-3
Edizione	[2nd ed.]
Descrizione fisica	1 online resource (425 p.)
Disciplina	005.14
Soggetti	Debugging in computer science Data editing
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Previous ed.: Amsterdam; London: Morgan Kaufmann, 2005.
Nota di bibliografia	Includes bibliographical references and index.
Nota di contenuto	Front Cover; Title Page; Copyright Page; Table of Contents; Foreword; Preface; Chapter 1. How Failures Come to Be; 1.1 My Program Does Not Work!; 1.2 From Defects to Failures; 1.3 Lost in Time and Space; 1.4 From Failures to Fixes; 1.4.1 Track the Problem; 1.4.2 Reproduce the Failure; 1.4.3 Automate and Simplify the Test Case; 1.4.4 Find Possible Infection Origins; 1.4.5 Focus on the Most Likely Origins; 1.4.6 Isolate the Origin of the Infection; 1.4.7 Correct the Defect; 1.5 Automated Debugging Techniques; 1.6 Bugs, Faults, or Defects?; 1.7 Concepts; How to debug a program; 1.8 Tools 1.9 Further ReadingExercises; Chapter 2. Tracking Problems; 2.1 Oh! All These Problems; 2.2 Reporting Problems; 2.2.1 Problem Facts; 2.2.2 Product Facts; 2.2.3 Querying Facts Automatically; 2.3 Managing Problems; 2.4 Classifying Problems; 2.4.1 Severity; 2.4.2 Priority; 2.4.3 Identifier; 2.4.4 Comments; 2.4.5 Notification; 2.5 Processing Problems; 2.6 Managing Problem Tracking; 2.7 Requirements as Problems; 2.8 Managing Duplicates; 2.9 Relating Problems and Fixes; 2.10 Relating Problems and Tests; 2.11 Concepts; How to obtain the relevant problem information How to write an effective problem reportHow to organize the debugging process; How to track requirements; How to keep problem

tracking simple; How to restore released versions; How to separate fixes and features; How to relate problems and fixes; How to relate problems and tests, make a problem report obsolete; 2.12 Tools; 2.13 Further Reading; Exercises; Chapter 3. Making Programs Fail; 3.1 Testing for Debugging; 3.2 Controlling the Program; 3.3 Testing at the Presentation Layer; 3.3.1 Low-Level Interaction; 3.3.2 System-Level Interaction; 3.3.3 Higher-Level Interaction
3.3.4 Assessing Test Results3.4 Testing at the Functionality Layer; 3.5 Testing at the Unit Layer; 3.6 Isolating Units; 3.7 Designing for Debugging; 3.8 Preventing Unknown Problems; 3.9 Concepts; How to test for debugging; How to automate program execution; How to test at the presentation layer; How to test at the functionality layer; How to test at the unit layer; How to isolate a unit; How to design for debugging; How to prevent unknown problems; 3.10 Tools; 3.11 Further Reading; Exercises; Chapter 4. Reproducing Problems; 4.1 The First Task in Debugging
4.2 Reproducing the Problem Environment4.3 Reproducing Program Execution; 4.3.1 Reproducing Data; 4.3.2 Reproducing User Interaction; 4.3.3 Reproducing Communications; 4.3.4 Reproducing Time; 4.3.5 Reproducing Randomness; 4.3.6 Reproducing Operating Environments; 4.3.7 Reproducing Schedules; 4.3.8 Physical Influences; 4.3.9 Effects of Debugging Tools; 4.4 Reproducing System Interaction; 4.5 Focusing on Units; 4.5.1 Setting Up a Control Layer; 4.5.2 A Control Example; 4.5.3 Mock Objects; 4.5.4 Controlling More Unit Interaction; 4.6 Reproducing Crashes; 4.7 Concepts
How to reproduce the problem

Sommario/riassunto

This book is proof that debugging has graduated from a black art to a systematic discipline. It demystifies one of the toughest aspects of software programming, showing clearly how to discover what caused software failures, and fix them with minimal muss and fuss. The fully updated second edition includes 100+ pages of new material, including new chapters on Verifying Code, Predicting Erors, and Preventing Errors. Cutting-edge tools such as FindBUGS and AGITAR are explained, techniques from integrated environments like Jazz.net are highlighted, and all-new demos with ESC/Java and Spec#