

1. Record Nr.	UNINA9910143577203321
Autore	Laird Linda M. <1952->
Titolo	Software measurement and estimation : a practical approach // Linda M. Laird, M. Carol Brennan
Pubbl/distr/stampa	Hoboken, New Jersey : , : John Wiley & Sons, , 2006 [Piscataway, New Jersey] : , : IEEE Xplore, , [2006]
ISBN	1-280-46844-0 9786610468447 0-470-24780-0 0-471-79253-5 0-471-79252-7
Descrizione fisica	1 online resource (276 p.)
Collana	Quantitative software engineering series ; ; 2
Altri autori (Persone)	BrennanM. Carol <1954->
Disciplina	005.1 005.14
Soggetti	Software measurement Software engineering
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Description based upon print version of record.
Nota di bibliografia	Includes bibliographical references and index.
Nota di contenuto	Acknowledgments -- 1. Introduction -- 1.1 Objective -- 1.2 Approach -- 1.3 Motivation -- 1.4 Summary -- References -- Chapter 1 Side Bar -- 2. What to Measure -- 2.1 Method 1: The Goal Question Metrics Approach -- 2.2 Extension to GQM: Metrics Mechanism is Important -- 2.3 Method 2: Decision Maker Model -- 2.4 Method 3: Standards Driven Metrics -- 2.5 What to Measure is a Function of Time -- 2.6 Summary -- References -- Exercises -- Project -- 3. Fundamentals of Measurement -- 3.1 Initial Measurement Exercise -- 3.2 The Challenge of Measurement -- 3.3 Measurement Models -- 3.3.1 Text Models -- 3.3.2 Diagrammatic Models -- 3.3.3 Algorithmic Models -- 3.3.4 Model Examples: Response Time -- 3.3.5 The Pantometric Paradigm - How to Measure Anything -- 3.4 Meta-Model for Metrics -- 3.5 The Power of Measurement -- 3.6 Measurement Theory -- 3.6.1 Introduction to Measurement Theory -- 3.6.2 Measurement Scales -- 3.6.3 Measures of Central Tendency and Variability -- 3.6.3.1 Measures of Central Tendency -- 3.6.3.2 Measures of Variability -- 3.6.4 Validity

and Reliability of Measurement -- 3.6.5 Measurement Error -- 3.7 Accuracy versus Precision and the Limits of Software Measurement -- 3.7.1 Summary -- 3.7.2 Problems -- 3.7.3 Project -- References -- 4. Measuring the Size of Software -- 4.1 Physical Measurements of Software -- 4.1.1 Measuring Lines of Code -- 4.1.1.1 Code Counting Checklists -- 4.1.2 Language Productivity Factor -- 4.1.3 Counting Reused and Refactored Code -- 4.1.4 Counting Non-Procedural Code Length -- 4.1.5 Measuring the Length of Specifications and Design -- 4.2 Measuring Functionality -- 4.2.1 Function Points -- 4.2.1.1 Counting Function Points -- 4.2.2 Function Point Counting Exercise -- 4.2.3 Converting Function Points to Physical Size -- 4.2.4 Converting Function Points to Effort -- 4.2.5 Other Function Point Engineering Rules -- 4.2.6 Function Point Pros and Cons -- 4.3 Feature Points -- 4.4 Size Summary -- 4.5 Size Exercises -- 4.6 Theater Tickets Project. References -- 5. Measuring Complexity -- 5.1 Structural Complexity -- 5.1.1 Size as a Complexity Measure -- 5.1.1.1 System Size and Complexity -- 5.1.1.2 Module Size and Complexity -- 5.1.2 Cyclomatic Complexity -- 5.1.3 Halstead's Metrics -- 5.1.4 Information Flow Metrics -- 5.1.5 System Complexity -- 5.1.5.1 Maintainability Index -- 5.1.5.2 The Agresti-Card System Complexity Metric -- 5.1.6 Object-Oriented Design Metrics -- 5.1.7 Structural Complexity Summary -- 5.2 Conceptual Complexity -- 5.3 Computational Complexity -- 5.4 Complexity Metrics Summary -- 5.5 Complexity Exercises -- 5.6 Projects -- References -- 6. Estimating Effort -- 6.1 Effort Estimation - Where are we? -- 6.2 Software Estimation Methodologies and Models -- 6.2.1 Expert Estimation -- 6.2.1.1 Work and Activity Decomposition -- 6.2.1.2 System Decomposition -- 6.2.1.3 The Delphi Methods -- 6.2.2 Using Benchmark Size Data -- 6.2.2.1 Lines of Code Benchmark Data -- 6.2.2.2 Function Point Benchmark Data -- 6.2.3 Estimation by Analogy -- 6.2.3.1 Traditional Analogy Approach -- 6.2.3.2 Analogy Summary -- 6.2.4 Proxy Point Estimation Methods -- 6.2.4.1 Meta-Model for Effort Estimation -- 6.2.4.2 Function Points -- 6.2.4.2.1 COSMIC Function Points -- 6.2.4.3 Object Points -- 6.2.4.4 Use Case Sizing Methodologies -- 6.2.4.4.1 Use Case Points Methodology -- 6.2.4.4.2 Example: Use Case Point Methodology Example: Home Security System -- 6.2.4.4.3 Use Case Point Methodology Effectiveness -- 6.2.5 Custom Models -- 6.2.6 Algorithmic Models -- 6.2.6.1 Manual Models -- 6.2.6.2 Estimating Project Duration -- 6.2.6.3 Tool Based Models -- 6.3 Combining Estimates -- 6.4 Estimating Issues -- 6.4.1 Targets vs. Estimates -- 6.4.2 The Limitations of Estimation - Why? -- 6.4.3 Estimate Uncertainties -- 6.5 Estimating Early and Often -- 6.6 Estimation Summary -- 6.7 Estimation Problems -- 6.8 Estimation Project - Theater Tickets -- References -- 7. In Praise of Defects: Defects and Defect Metrics -- 7.1 Why study and measure defects?. 7.2 Faults vs. failures -- 7.3 Defect Dynamics and Behaviors -- 7.3.1 Defect Arrival Rates -- 7.3.2 Defects vs. Effort -- 7.3.3 Defects vs. Staffing -- 7.3.4 Defect Arrival Rates vs. Code Production Rate -- 7.3.5 Defect Density vs. Module Complexity -- 7.3.6 Defect Density vs. System Size -- 7.4 Defect Projection Techniques and Models -- 7.4.1 Dynamic Defect Models -- 7.4.1.1 Rayleigh Models -- 7.4.1.2 Exponential and S-Curves Arrival Distribution Models -- 7.4.1.3 Empirical Data and Recommendations for Dynamic Models -- 7.4.2 Static Defect Models -- 7.4.2.1 Defect Insertion and Removal Model -- 7.4.2.2 Defect Removal Efficiency - A Key Metric -- 7.4.2.3 Static Defect Model Tools -- 7.5 Additional Defect Benchmark Data -- 7.5.1 Defect Data By Application Domain -- 7.5.2 Cumulative Defect Removal Efficiency (DRE) Benchmark -- 7.5.3 SEI Levels and Defect Relationships -- 7.5.4 Latent Defects -- 7.5.5 Other Defects Benchmarks and a Few

Recommendations+ -- 7.6 Cost Effectiveness of Defect Removal by Phase -- 7.7 Defining and Using Simple Defect Metrics: An example -- 7.8 Some Paradoxical Patterns for Customer Reported Defects -- 7.9 Defect Summary -- 7.10 Problems -- 7.11 Projects -- 7.12 Answers to the initial questions -- References -- 8. Software Reliability Measurement and Prediction -- 8.1 Why study and measure software reliability? -- 8.2 What is reliability? -- 8.3 Faults and failures -- 8.4 Failure Severity Classes -- 8.5 Failure Intensity -- 8.6 The Cost of Reliability -- 8.7 Software Reliability Theory -- 8.7.1 Uniform and Random Distributions -- 8.7.2 The probability of failure during a time interval -- 8.7.3  $F(t)$  - The Probability of Failure by time  $t$  -- 8.7.4  $R(t)$  - The Reliability Function -- 8.7.5 Reliability Theory Summarized -- 8.8 Reliability Models -- 8.8.1 Types of Models -- 8.8.2 Predicting Number of Defects Remaining -- 8.8.3 Reliability Growth Models -- 8.8.4 Model Summary -- 8.9 Failure Arrival Rates -- 8.9.1 Predicting Failure Arrival Rates Using Historical Data. -- 8.9.2 Engineering Rules for MTTF -- 8.9.3 Musa's Algorithm -- 8.9.4 Operational Profile Testing -- 8.9.5 Predicting Reliability Summary -- 8.10 But when do I ship? -- 8.11 System Configurations: Probability and Reliability -- 8.12 Answers to Initial Question -- 8.13 Reliability Summary -- 8.14 Reliability Exercises -- 8.15 Reliability Project -- References -- 9. Response Time and Availability -- 9.1 Response Time Measurements -- 9.2 Availability -- 9.2.1 Availability Factors -- 9.2.2 Outage Scope -- 9.2.3 Complexities in Measuring Availability -- 9.2.4 Software Rejuvenation -- 9.2.4.1 Software Aging -- 9.2.4.2 Classification of Faults -- 9.2.4.3 Software Rejuvenation Techniques -- 9.2.4.4 Impact of Rejuvenation on Availability -- 9.3 Summary -- 9.4 Problems -- 9.5 Project -- References -- 10. Measuring Progress -- 10.1 Project Milestones -- 10.2 Code Integration -- 10.3 Testing Progress -- 10.4 Defects Discovery and Closure -- 10.4.1 Defect Discovery -- 10.4.2 Defect Closure -- 10.5 Process Effectiveness -- 10.6 Summary -- References -- Problems -- 11. Outsourcing -- 11.1 The "O" Word -- 11.2 Defining Outsourcing -- 11.3 Risks Management and Outsourcing -- 11.4 Metrics and the Contract -- 11.5 Summary -- References -- Exercises -- Problems -- Chapter 11 Sidebar -- 12. Financial Measures for the Software Engineer -- 12.1 It's All About the Green -- 12.2 Financial Concepts -- 12.3 Building the Business Case -- 12.3.1 Understanding Costs -- 12.3.1.1 Salaries -- 12.3.1.2 Overhead Costs -- 12.3.1.3 Risk Costs -- 12.3.1.3.1 Identifying Risk -- 12.3.1.3.2 Assessing Risks -- 12.3.1.3.3 Planning for Risk -- 12.3.1.3.4 Monitoring Risk -- 12.3.1.4 Capital versus Expense -- 12.3.2 Understanding Benefits -- 12.3.3 Business Case Metrics -- 12.3.3.1 Return on Investment -- 12.3.3.2 Pay-Back Period -- 12.3.3.3 Cost/Benefit Ratio -- 12.3.3.4 Profit & Loss Statement -- 12.3.3.5 Cash Flow -- 12.3.3.6 Expected Value -- 12.4 Living the Business Case -- 12.5 Summary -- References -- Problems. -- 13. Benchmarking -- 13.1 What is Benchmarking -- 13.2 Why Benchmark -- 13.3 What to Benchmark -- 13.4 Identifying and Obtaining a Benchmark -- 13.5 Collecting Actual Data -- 13.6 Taking Action -- 13.7 Current Benchmarks -- 13.8 Summary -- References -- Problems -- Projects -- 14. Presenting Metrics Effectively to Management -- 14.1 Decide on the Metrics -- 14.2 Draw the Picture -- 14.3 Create a Dashboard -- 14.4 Drilling for Information -- 14.5 Example for the Big Cheese -- 14.6 Evolving Metrics -- 14.7 Summary -- References -- Problems -- Project -- Index.

of the required effort and the resulting quality of a software project.

2. Record Nr.	UNISALENTO991000808059707536
Autore	Hilgard, Ernest R.
Titolo	Psicologia : corso introduttivo / Ernest R. Hilgard ; traduzione di M. Chiarenza e R. Tettucci ; a cura di Gabriele Noferi
Pubbl/distr/stampa	[Firenze] : Giunti-Barbèra, c1971
Descrizione fisica	787 p. : ill. ; 25 cm
Collana	Biblioteca internazionale di psicologia
Altri autori (Persone)	Chiarenza, Mario Tettucci, Roberto Noferi, Gabriele
Disciplina	150.1
Soggetti	Psicologia - Manuali
Lingua di pubblicazione	Italiano
Formato	Materiale a stampa
Livello bibliografico	Monografia