

EXHIBIT M

Contents

| | |
|---|-----------|
| Foreword: Under the Tip of the Estimation Iceberg <i>by Barry Boehm</i> | xvii |
| Foreword <i>by Donald J. Reifer</i> | xix |
| Preface | xxi |
| How This Book Came about from a Galorath Viewpoint..... | xxi |
| Audience | xxiii |
| Structure of the Book | xxv |
| What Can You Expect from the Book?..... | xxv |
| Acknowledgments..... | xxvii |
| The Authors..... | xxix |
| 1 The Problem | 1 |
| Introduction..... | 1 |
| Focus of the Book..... | 4 |
| Why Software Projects Fail..... | 4 |
| Why Software Projects Fail: Problems with Estimation..... | 7 |
| Why Software Projects Fail: Size Estimates | 9 |
| Why Estimates Fail..... | 13 |
| Historical Data..... | 14 |
| Overly Optimistic Leadership and Management..... | 16 |
| Failure to Use Estimate..... | 16 |
| Failure to Keep Estimate Current | 18 |
| Role of Risk Management in Estimating..... | 18 |
| The Solution: Software Estimation — Ten-Step Process | 20 |
| Summary | 21 |
| Endnotes..... | 21 |
| 2 Introduction to Software Estimation Techniques and | |
| Estimate Planning..... | 25 |
| Introduction and Chapter Goals..... | 25 |
| Need for Efficient Software Project Management Metrics..... | 26 |
| Core Metrics Categories..... | 28 |

Software Project Estimates: Foundations of Software Project Management 30

Software Estimation Concepts..... 34

Project Estimation Process..... 35

 Step One: Establish Estimate Scope and Purpose 36

 Step Two: Establish Technical Baseline, Ground Rules, and Assumptions..... 38

 Step Three: Collect Data 39

 Underlying Information 39

 Interview with Judy Galorath 45

 Software Data Collection Process..... 48

 Software Data Collection Lessons Learned..... 50

 Prioritizing Estimation Effort 54

 Summary 54

Endnotes..... 54

3 Executing the Estimate 57

 Introduction and Chapter Goal..... 57

 Step Four: Software Sizing..... 57

 Predicting Size 58

 Size Estimation Approaches 60

 Deciding on a Metric..... 61

 When to Use SLOC..... 63

 When to Use Function Points..... 63

 Steps to Estimating Software Size..... 64

 Sizing Step 1: Baseline Definition of the Size Metric You Will Use... 65

 Sizing Step 2: Define Sizing Objectives .. 65

 Sizing Step 3: Plan Data and Resource Requirements... 66

 Sizing Step 4: Identify and Evaluate Software Requirements..... 66

 Sizing Step 5: Use Several Independent Techniques and Sources... 66

 Sizing Step 6: Tracking..... 68

 Sizing Databases..... 68

 Legacy Software Rework 69

 Sizing Number of Functions to Be Learned, Used, and Integrated for COTS 70

 Step Five: Prepare Baseline Estimate..... 70

 Software Productivity Laws 72

 Bottom-Up Estimating..... 78

 Software Cost Models 78

 Organizing the Estimating Process 86

 Delphi and Wideband Delphi..... 87

 Activity-Based Estimates 89

 Step Six: Quantify Risks and Risk Analysis 99

 Cost Estimation Risks..... 99

 Summary 102

Endnotes..... 102

| | |
|---------|--|
| 30 | 4 Planning and Controlling the Project via the Estimate105 |
| 34 | Introduction 105 |
| 35 | Step Seven: Estimate Validation and Review..... 105 |
| 36 | Estimate Review Process 107 |
| 38 | Estimate Review Activities 108 |
| 39 | Cost per Unit of Code Developed 109 |
| 39 | Magic Bullets (Otherwise Known as Technical Leaps) 109 |
| 45 | Unrealistic Schedules 110 |
| 48 | Inaccurate Sizing 110 |
| 50 | Complexity versus Risk..... 111 |
| 54 | Careful Evaluation of Preexisting and COTS Software 112 |
| 54 | Off-the-Shelf Integration 112 |
| ..57 | Function Point Counting Checklist 113 |
| ..57 | Sanity Counts 113 |
| ..57 | Lack of Convergence 113 |
| ..58 | Double Counting..... 113 |
| ..60 | Sample and Statistical Concerns 114 |
| ..61 | Probability Level..... 114 |
| ..63 | Falsely Bounded Risk 114 |
| ..64 | Costs 115 |
| ..65 | Are Staff Costs Fully Burdened?..... 115 |
| ..65 | How Many Hours Are in a Staff Month? 115 |
| ..66 | Staff and Effort Accounting..... 115 |
| ..66 | Does Overtime Count?..... 115 |
| ..68 | What Level of Management Participates? 115 |
| ..68 | How Efficiently Is Staff Allocated?..... 116 |
| ..69 | Are Experience Levels Honestly Rated? 116 |
| ..70 | Schedules 116 |
| ..70 | What Is the Proportion of Daily Billable Work Done?..... 116 |
| ..72 | Will Development Have Lags? 116 |
| ..78 | If Several Software Elements Are Developed, How Are They Scheduled? 116 |
| ..86 | Is It More Important to Save Time or Staff Cost?..... 117 |
| ..87 | Sanity Check 117 |
| ..89 | Estimate Process Questionnaire 117 |
| ..99 | Step Eight: Generate Project Plan 128 |
| ..99 | Action Items by Project Phase 129 |
| 02 | Determining Costs from Effort Estimates 133 |
| 02 | Estimating Personnel Mix 133 |
| 02 | Labor Proportions..... 134 |
| 02 | Other Costs 134 |
| 02 | Travel Costs 134 |
| 02 | Personnel Costs 134 |
| 02 | Depreciation Costs 137 |
| 02 | Training Costs 137 |

x ■ *Software Sizing, Estimation, and Risk Management*

| | |
|---|------------|
| Independent Verification and Validation or Independent Quality Assurance..... | 137 |
| Inflation..... | 137 |
| Overhead..... | 138 |
| Estimating Schedule in Calendar Months..... | 138 |
| Effect of Management and Process on Estimates | 138 |
| Impact of Software Project Management on Software Development Plan | 138 |
| Effect of Software Processes on Software Development Plan..... | 141 |
| Step Nine: Document Estimate and Lessons Learned | 143 |
| Conducting Lessons-Learned Review | 144 |
| Cause Segment..... | 145 |
| Effects Segment..... | 145 |
| Modeling Improvement Segment..... | 146 |
| Step Ten: Track Project throughout Development | 146 |
| Refining Estimates throughout Project | 146 |
| Summary | 147 |
| Endnotes..... | 148 |
| | |
| 5 Source Lines of Code | 149 |
| Introduction..... | 149 |
| Terminology and Definitions | 150 |
| SLOC Realities and Risks..... | 152 |
| Using SLOC | 153 |
| Logical SLOC Counting Details..... | 155 |
| Logical SLOC Detailed Definitions | 155 |
| Executable Statements | 155 |
| Data Declaration Statements | 158 |
| Compiler Directives..... | 158 |
| Line Counting Example | 161 |
| Estimation versus Counting SLOC | 162 |
| SLOC Considerations for Sizing Databases | 162 |
| Language Impact on Size Conversion..... | 163 |
| Effective Size | 164 |
| Productivity Based on Effective Size..... | 164 |
| Accounting for SLOC Growth..... | 164 |
| Estimating Size Growth Conclusions..... | 166 |
| Finding Automated Code Counters for Existing Systems..... | 167 |
| Pros and Cons of SLOC | 169 |
| Arguments against Use of Lines of Code as Sizing Metric | 170 |
| Risks Resulting from Using SLOC to Estimate..... | 170 |
| Risk Management and Control of SLOC Estimates | 171 |
| Summary..... | 171 |

ality
 137
 137
 138
 138
 138
 cent
 138
 141
 143
 144
 145
 145
 146
 146
 146
 147
 148
 149
 149
 150
 152
 153
 155
 155
 155
 158
 158
 161
 162
 162
 163
 164
 164
 164
 166
 167
 169
 170
 170
 171
 171

SEI Checklist..... 172
 SEI Definition Checklist for Source Statement Counts 172
 Codes for Various Programming Languages..... 176
 Endnotes..... 185

6 Function-Based Sizing 187

Introduction 187
 Origins and History of Functional Metrics 188
 ISO Involvement 190
 International Function Point User Group Counting Standards: Basic
 Process Definition 191
 IFPUG Definitions 192
 IFPUG Steps 192
 Step 1: Determine Type of Function Point Count..... 192
 Step 2: Determine Application Boundary..... 194
 Step 3: Identify Functional Categories ... 195
 External Input (EI) 196
 External Output (EO)..... 199
 External Inquiry (EQ) 201
 External Interface File (EIF) 204
 Internal Logical File (ILF)..... 206
 Step 4: Count Data Functions (ILFs and EIFs)..... 209
 Step 5: Count Transactional Functions (EIs, EOs, and EQs) 210
 Step 6: Evaluate Value Adjustment Factors..... 211
 Step 7: Compute Unadjusted and Adjusted Function Point Counts... 222
 SEER-Function-Based Sizing (SEER-FBS)..... 224
 SEER-FBS External Inputs (EIs) 225
 SEER-FBS Subcategories for External Inputs 226
 Rating Complexity for External Inputs..... 226
 SEER-FBS External Outputs (EOs)..... 226
 SEER-FBS Subcategories for External Outputs..... 227
 Rating Complexity for External Outputs 227
 SEER-FBS External Inquiries (EQs)..... 227
 Rating Complexity for External Inquiries..... 228
 SEER-FBS Subcategories for External Inquiries 228
 SEER-FBS External Interface Files (EIFs) 228
 SEER-FBS Subcategories for External Interface Files 228
 Rating Complexity for External Interface Files..... 229
 SEER-FBS Internal Logical Files (ILFs) 229
 SEER-FBS Subcategories for Internal Logical Files..... 229
 Rating Complexity for Internal Logical Files 229
 SEER-FBS Extended Category: Internal Functions..... 230
 Effective Function Points..... 230
 Using Function Points..... 233
 Early Function Point Counting (Estimating) 236

| | |
|---|------------|
| Analysis of Function Point Rules in Tree-Based Framework..... | 236 |
| Description of Tree and Results | 237 |
| Backfiring | 237 |
| Possible Errors in Function Point Counting..... | 239 |
| Pros and Cons of Function Points | 240 |
| Pros of Function Points | 240 |
| Cons of Function Points | 241 |
| When to Use Function Points..... | 242 |
| Function Point Risk Management..... | 242 |
| Function Point Counting Risk Checklist | 243 |
| Summary | 243 |
| Endnotes..... | 251 |
| 7 Object-Oriented Sizing: Object and Use-Case Sizing..... | 253 |
| Introduction..... | 253 |
| Background of Object-Oriented Design | 254 |
| Overview of Object-Oriented Techniques | 255 |
| Object Points | 256 |
| Performing Object Point Counts..... | 256 |
| Object Point Definitions | 256 |
| Classes..... | 256 |
| Services (Methods) | 259 |
| Predictive Object Points | 262 |
| Development of Use-Case Metric | 262 |
| Calculation of Unadjusted Use-Case Points..... | 263 |
| Adjustment of Use-Case Point Count (Optional) | 265 |
| Concluding Comments about Use-Case Points | 265 |
| Sizing Web Development | 265 |
| Risk Associated with Object-Oriented Projects | 267 |
| Summary..... | 272 |
| Endnotes..... | 273 |
| 8 Software Reuse and Commercial Off-the-Shelf Software | 275 |
| Introduction..... | 275 |
| Reusable Software..... | 277 |
| Integrating Commercial Off-the-Shelf Software | 279 |
| Fundamental Differences between COTS Software and Custom Development | 282 |
| Items Not Estimated as COTS..... | 283 |
| Weighing Use of COTS..... | 284 |
| Case Studies: Real-World Experiences with COTS..... | 284 |
| Case 1: Components Had Critical Defects and Were Modified by Developer | 284 |
| Case 2: Powerful (and Defect-Ridden) COTS Component..... | 285 |
| Case 3: Application Integrated (Loosely Coupled) without Problems... | 285 |
| Evaluating and Estimating COTS | 285 |
| Three Components of COTS Integration | 286 |

| | | |
|-----------|--|------------|
| 236 | Estimating COTS Integration..... | 287 |
| 237 | Using Function Points and Estimating Model Lacking COTS-Specific | |
| 237 | Capability..... | 287 |
| 239 | Integration of Stand-Alone COTS Software..... | 288 |
| 240 | Stand-Alone COTS Software with Significant Configuration..... | 288 |
| 240 | Using SEER-SEM Cost Drivers to Estimate COTS..... | 288 |
| 241 | Object Sizing..... | 291 |
| 242 | Feature Sizing..... | 291 |
| 242 | Rules of Thumb for COTS Integration..... | 293 |
| 243 | Experience with COTS Product..... | 293 |
| 243 | Scope of COTS..... | 293 |
| 251 | Evaluation and Selection of COTS Products..... | 294 |
| 253 | COTS Risks..... | 294 |
| 253 | Risk Reduction..... | 296 |
| 254 | Risks Associated with Reuse and COTS..... | 297 |
| 255 | Summary..... | 297 |
| 256 | Endnotes..... | 301 |
| 256 | 9 Performing to Estimate: Managing and Monitoring | |
| 256 | Development..... | 303 |
| 256 | Introduction..... | 303 |
| 259 | Metric Reporting..... | 304 |
| 262 | Metrics Sets..... | 309 |
| 262 | Productivity..... | 309 |
| 263 | Productivity Monitoring..... | 309 |
| 265 | Using Earned Value Management..... | 318 |
| 265 | When Reality Sets In..... | 323 |
| 265 | "Shoestring" Project Environments..... | 324 |
| 267 | Process Performance..... | 325 |
| 272 | Technology Solutions..... | 326 |
| 273 | Understanding Process Selection Constraints..... | 327 |
| 275 | Product Quality and Stability..... | 330 |
| 275 | Defects..... | 331 |
| 277 | Code Inspections..... | 333 |
| 279 | Staffing Levels..... | 335 |
| 279 | Team Performance..... | 335 |
| 282 | Summary..... | 337 |
| 283 | Endnotes..... | 337 |
| 284 | 10 Risk Management Process..... | 339 |
| 284 | Introduction..... | 339 |
| 284 | History of Risk Management..... | 340 |
| 284 | Cultural Obstacles to Managing Risk..... | 343 |
| 285 | Risks versus Problems..... | 345 |
| 285 | Risk Management Success Factors..... | 347 |
| 285 | Essential Risk Management Definitions..... | 349 |
| 286 | Introduction to Risk Management Concepts..... | 350 |

xiv ■ *Software Sizing, Estimation, and Risk Management*

| | |
|--|------------|
| Computing a Risk Index..... | 352 |
| Risk Management Processes..... | 356 |
| Seven Steps to Risk Management..... | 359 |
| Step 1: Establish Risk Policy, Obtain Commitment to Manage Risk, and Develop Plan | 359 |
| Risk Management Planning | 360 |
| "How-To" Procedures: Essential Planning Elements | 362 |
| Step 2: Designate Risk Officer | 368 |
| Risk Officer Case Study..... | 371 |
| Relationship of Risk Officer and Management..... | 371 |
| Step 3: Identify Risks | 372 |
| Risk Identification Techniques..... | 374 |
| Risk Characterization..... | 377 |
| Potential Risk Identification Activities during Estimation | 378 |
| Step 4: Risk Analysis | 381 |
| Use of Metrics | 382 |
| Use of Quantitative Triggers | 382 |
| Step 5: Prioritize Risks | 383 |
| Step 6: Report Risks | 384 |
| Reporting Problems versus Risks..... | 384 |
| Risk Reporting by Exposure..... | 385 |
| Step 7: Establish Risk Reserve | 386 |
| Basic Risk Management Rules | 387 |
| Risk Analysis Viewed as Uncertainty Analysis | 387 |
| Establishing Risk Reserve Using Commercial Grade Models | 388 |
| Risk Management Dealing with Cost Uncertainty | 388 |
| Risk Analysis at the Work Element Level | 389 |
| Pert Distribution Characteristics | 390 |
| Probability and Intuition..... | 391 |
| Probability-Based Risk Outputs | 392 |
| Project and Roll-Up Risk Calculation | 392 |
| Summary | 393 |
| Endnotes..... | 395 |
| | |
| 11 Applying SEER-SEM to Estimation Processes | 397 |
| Introduction to SEER-SEM Project Manager Edition Tools..... | 398 |
| Details and Uses | 401 |
| Summary Input and Output Definitions | 402 |
| SEER-SEM Concept..... | 403 |
| SEER-SEM Sizing | 405 |
| SEER-SEM Programmatic Architecture | 406 |
| Open Databases | 406 |
| Communicating with SEER-SEM via Microsoft COM | 407 |
| Server Mode..... | 407 |
| Applying SEER-SEM Project Manager Edition to the Estimation Process | 407 |

| | | |
|---------|---|-----|
| ... 352 | Steps 1 through 3: Establish Estimate Scope and Purpose; | |
| ... 356 | Establish Technical Baseline, Ground Rules, and Assumptions; and | |
| ... 359 | Collect Data | 407 |
| | SEER-SEM Software Sizing (Step 4)..... | 409 |
| ... 359 | Manual Sizing | 410 |
| ... 360 | Automated Sizing with SEER-AccuScope | 410 |
| ... 362 | Choosing Knowledge Bases for Reuse Estimation..... | 412 |
| ... 368 | Using SEER Function-Based Sizing for Size Estimates..... | 420 |
| ... 371 | Using Number of Programs Included in Size | 420 |
| ... 371 | SEER-SEM Estimation Process (Step 5) | 421 |
| ... 372 | SEER-SEM Estimation Process Step 5b: Select Knowledge Bases..... | 423 |
| ... 374 | SEER-SEM Estimation Process Step 5c: Specify Project Constraints..... | 424 |
| ... 377 | SEER-SEM Estimation Process Step 5d: Adjust Individual Parameters... | 425 |
| ... 378 | SEER-SEM Estimation Process Step 6: Quantify Risks and Risk | |
| ... 381 | Analysis | 426 |
| ... 382 | Distributions..... | 427 |
| ... 382 | Probability Distribution of Output Ranges..... | 428 |
| ... 383 | Risk Factor Analysis with Sensitivity Charts | 429 |
| ... 384 | Ranked Risks with Top Ten Cost Drivers Chart | 431 |
| ... 384 | Precise Estimate Distributions through Risk Analysis Report..... | 431 |
| ... 385 | SEER-SEM Estimation Process Step 7: Review, Verify, and Validate | |
| ... 386 | Estimate..... | 432 |
| ... 387 | SEER-SEM Estimation Process Step 8: Generate Project Plan | 434 |
| ... 387 | SEER-SEM Estimation Process Step 9: Document Estimate and | |
| ... 388 | Lessons Learned | 435 |
| ... 388 | Custom Knowledge Bases and Calibration..... | 435 |
| ... 389 | Calibration (Part of Lessons Learned) | 435 |
| ... 390 | Constructing Calibration Factors | 436 |
| ... 391 | SEER-SEM Estimation Process Step 10: Track Project..... | 436 |
| ... 392 | SEER-SEM Internals | 436 |
| ... 392 | SEER-SEM Basic Size Definition..... | 437 |
| ... 393 | SEER-SEM Staff Hour Definition | 437 |
| ... 395 | SEER-SEM Mathematical Model Overview | 437 |
| | Effective Size Mathematics..... | 437 |
| | Function-Based Sizing Mathematics..... | 442 |
| | Parameters..... | 442 |
| | Knowledge Bases..... | 442 |
| | Effective Technology Calculation..... | 443 |
| ... 397 | Effort, Schedule, and Staffing Calculations | 445 |
| ... 398 | Basic Definitions..... | 445 |
| ... 401 | Basic Effort and Schedule Equations..... | 445 |
| ... 402 | Optimal Effort Calculations | 446 |
| ... 403 | Relaxed Schedule Calculations..... | 447 |
| ... 405 | Applying Adjustment Factors | 448 |
| ... 406 | SEER-SEM Parameter Definitions | 448 |
| ... 406 | Contents | 448 |
| ... 407 | Sizing Parameters | 449 |
| ... 407 | | |

| | |
|--|------------|
| Technology and Environment Parameters | 453 |
| Commercial Off-the-Shelf (COTS) Parameters | 476 |
| Other Parameters | 484 |
| Summary | 496 |
| Endnotes | 497 |
| 12 SEER-SEM Solutions for Project Management and Control | 499 |
| Introduction | 499 |
| CMMI Process Areas for Project Management | 500 |
| Solution 1: Application of Basic SEER-SEM for Project Management and Control | 501 |
| Solution 2: SEER-SEM Client for Microsoft Project | 503 |
| Using the Client for Detailed Project Planning | 504 |
| Solution 3: SEER-PPMC (Parametric Project Monitoring and Control) | 506 |
| Implementing Planning and Control Process with SEER-PPMC | 510 |
| Earned Value Metrics and Calculations Used in SEER-PPMC | 512 |
| Summary | 518 |
| Endnotes | 518 |
| Index | 519 |

For
of t

Many pe
use invol
resulting
breakdown
Howev
running th
essential t
include:

- Id
mi
inc
wt
co
co
- De
po
ful
cat
- Us
sch
are
- Id
Th
the
ust
the

Table 3.1 (continued) Estimation Measures

| Issue | Source Lines of Code | Function Points |
|--|--|--|
| Postmortem use | Useful; easy to compare before-and-after results | Useful: easy to compare before-and-after results; easy to see exactly where variation occurred |
| Expense of estimate | Low: rapid and very inexpensive | Moderate to high; slower and potentially expensive |
| Ability to estimate with automated tools | High: tools such as SEER-AccuScope can estimate lines with relative ease | High; tools such as SEER-AccuScope can estimate function points with relative ease |

When to Use SLOC

SLOC has been the dominant method for sizing complicated, real time or embedded systems and works well for hand-generated systems in general. Use lines of code when SLOC-based historical data exists, when the development organization is comfortable with SLOC estimates, when additions to existing systems allow counting of actual SLOC in a system, and as a relatively easy check on other methods.

The great strength of SLOC is that it is easy to obtain. All other factors aside, it remains a fairly accurate predictor of development effort. By comparing code counts from past projects against a “rough order of magnitude” estimate for a proposed project, you can gain your first real understanding of project scope. By pairing SLOC estimates with other development factors, you will generally have enough information to develop a reliable estimate.

SLOC counts provide a firm indication of the volume of software generated, which is a first critical step for making comparisons and predictions. Despite the dominance of SLOC measures, some confusion exists regarding which types of lines to count, which has led to difficulty in comparing methods of counting SLOC. However, within the past several years, code counting methods have become more standardized. See Chapter 5 for a detailed discussion of lines of code.

When to Use Function Points

It is best to develop estimates based on function points when your project is largely comprised of information technology and the system’s functions

are adequately specified. Alternatively, you can estimate a function point count using other means such as SEER-AccuScope to estimate size. In addition, you should use function points when sizing by SLOC could be misleading. For example, code generators can automatically generate many lines of code, which makes the number of lines generated an unreliable predictor of the amount of effort required. The great strength of function point counts is that they are developed directly from specifications, independent of implementation, which means estimates of project scope are more comparable across projects.

Counting function points is a sophisticated method that cannot be done automatically. There are few shortcuts; you must ensure that it will be done properly by assigning adequately trained and experienced personnel. If you have counts from previous similar projects, be sure to study those counts carefully to ensure that the work performed on your current project is consistent with the method used on those projects. Although experienced counters can accomplish this method fairly quickly and efficiently, it is very important that the counts be performed correctly and consistently.

The function point counting process should be put in perspective. An experienced function point counter is usually able to count project specifications amounting to about 600 function points per day. As a typical MIS database project of this size might take a bit more than a year to develop and consume a bit more than a hundred person-months of effort, using function points does not necessarily involve a big investment in up-front planning. Many large organizations therefore keep a function point counter on staff or hire outside consultants when necessary. By using the SEER function-based sizing method, function points can be estimated without conducting detailed counts. Finally, it is important to understand that a combination of SLOC and function-based sizing can be the most appropriate way of estimating the size of an existing system to which functional enhancements are being made.

Steps to Estimating Software Size

Managers of software development projects are responsible for ascertaining progress, risk, productivity, and a host of other factors that are critical to success of their projects. Two important factors are the size of the product and the subsequent effort that will be required to develop it. If you want to contain the risk of unexpected cost growth for your project, it is essential that you use a software sizing method that is consistent and repeatable.

In order to usefully apply the general concepts and techniques of managing risk to your software engineering project, it is also essential that you regularly reestimate the size of the product and the associated cost of the project as project conditions change or product specifications

les versus
n module.
with the
of different
ment this

ating esti-
de will be
st complex
l it require

ask the set

rm will

endor is

associated

er greater

er may have
alyzed.

Function Point Counting Checklist

When presented with a function point count, carefully consider the following information.

Sanity Counts

Function point counts vary with counters. It is therefore useful to have a second counter double check (conduct a sanity count) of some part of a count before the entire count has been completed. By doing this, counting methodology questions can be resolved early. Alternately, using a sizing model can approximate the count quickly.

Lack of Convergence

Unless a counter is very experienced, a function point count should be conducted over several iterations. The more experience a counter has, the better he or she will understand the technique and be able to converge on a reliable number.

Double Counting

Keep careful track of requirements and the resulting function point counts to make sure that counts have not been replicated. The following are some typical sources of double counting:

- Referencing a File More than Once
Files should be counted in relation to the boundary of the entire application. For example, an external interface file is created completely outside the application boundary. On the other hand, an internal logical file is created completely within the application boundary. It should be clear that external and internal logical files are counted only once in a particular application, regardless of how often they are used.
- Confusing Designer's and User's Perspectives
Counting occurs from the user's perspective, which means that certain architectural details may remain hidden from the count and will legitimately not be counted. Another source of perspective-related confusion is the difference between physical and logical files. Sometimes, what the user sees as one logical file may actually reside in several physical files, while the opposite may also be

true. In either case, conduct the count as the user would see the components.

Sample and Statistical Concerns

When validating estimates based on project histories, be sure to assess the samples and statistics used and their associated risks.

Probability Level

The acquisition type of the project can determine the probability level of the estimate. When comparing contractor estimates with your own, be aware that a contractor's acceptable risk varies with the type of contract. A contractor may choose an estimate other than at the 50 percent probability. (Internal developments may also have the same issues with the use of probability.)

On *cost plus* jobs, a developer's overriding interest is in winning the contract, and so he may offer a more daring estimate. An estimate probability of 40 percent and even 30 percent may be chosen and could lead to a 60 or 70 percent chance of a cost overrun.

On a *fixed price* award, a developer bears the expense of a cost overrun and so is more fearful of bidding lower than practical. For such projects, an estimate probability of 60 percent or even 80 percent might be chosen.

Compare your estimate range and the actual bid. If the bid lies at the probability level in your estimate range that would be predicted by the type of contract, then the estimates probably agree. You can sometimes directly ask a contractor what probability level was chosen, but this information may or may not be available to you.

Falsely Bounded Risk

A risk analysis may be populated with engineers' and estimators' assumptions that may not admit the full range of possible outcomes. One useful countermeasure is comparing assumptions from different people and program components (such as separate computer programs) so that you can reconcile inconsistencies.

Bias

You must be careful to obtain a balanced sample, one that is not biased in any way, and particularly by factors that are unrelated to the project you are analyzing. Sample bias is a particular problem with small samples.

Try to obtain a sample with characteristics that are consistent with the project you are estimating.

Outliers

When assembling a sample to help you validate the estimate of a current project, some values can lie well outside the common range; these are called outliers. Outliers must be separately examined. An outlier may be an unrepresentative event that should be ignored or it may offer special lessons. An outlier that is valuable should not necessarily be included in the main sample and in the sample statistics. It may instead be used to offer instructive lessons outside the conventional statistical analysis.

Costs

Are Staff Costs Fully Burdened?

Ask the developer what its staff costs are, and make certain that the monthly staff rate has been agreed to. Understand whether costs are fully burdened or whether additional charges will be incurred; if so, ensure that these costs are included in the complete cost estimate.

How Many Hours Are in a Staff Month?

A common United States standard is 152 hours per staff month. If a developer's hours per month vary from what the estimate uses, you must normalize the results to make a viable comparison. Multiply all effort-month figures by the developer's hours per month and then divide this number by the hours per month that you are using. (Automated cost models like SEER-SEM do this automatically.)

Staff and Effort Accounting

Does Overtime Count?

If the development staff is planning to use unreported overtime, this will cause variation from the estimate.

What Level of Management Participates?

Account for labor directly applied to this project only. Confirm that the developer has not included upper level (executive) management.