PART NINE

Supplemental Information

This part of *Practical Software and Systems Measurement: A Foundation for Objective Project Management* provides supplemental information to promote a better understanding of the concepts and terms used within the Guide. It also provides additional background information about Practical Software and Systems Measurement.

The information in this part of the Guide includes:

- **Glossary** - This section provides definitions of terms used throughout the Guide.
- **List of Acronyms** - Acronyms used throughout the Guide are defined in this section.
- **Bibliography** - References related to PSM and software measurement are provided in this section.
- **PSM Project Information** - The project attribution policy and contact information are included in this section.
- **Comment Form** - Comments and suggestions concerning the PSM Guide may be submitted using this form.

[This page intentionally left blank.]

TABLE OF CONTENTS

[This page intentionally left blank.]

Glossary

actual data — See ***measured value***.

acquirer — The individual or organization responsible for funding or purchasing the software product or system. The acquirer can be defined as the customer for the software development or operations and maintenance effort. See ***customer***.

aggregation structure — Effective measurement analysis and reporting requires that the data be aggregated to higher levels of the product and organizational structure. The aggregation structure defines the different ways the measurement data can be grouped and organized for reporting on the project. The aggregation structure describes how the measurement data relates to existing product and process structures. This organization allows the measurement results to be combined, and later decomposed, meaningfully.

application software — Software produced for a specific user need, as opposed to general system software. Examples include software for navigation, fire control, payroll, or general ledger.

apply measures — In the PSM process, this term refers to one of the four basic measurement activities which comprise the software measurement process. The application activity involves collecting, analyzing, and reporting the measurement data. See ***tailor measures, implement process, evaluate measurement***.

attribute — Characteristics or properties assigned to each software measurement data item. Attributes provide categories for sorting and selecting data. The attributes of the measurement data must be collected with the data. An example of an attribute is problem reports priority levels, or development organization.

Capability Maturity Model (CMM) — Contains the essential elements of effective processes for one or more disciplines. These elements are organized into hierarchical stages or levels.

Commercial Off The Shelf (COTS) — Commercial items that require little or no unique government modifications or maintenance over the life cycle of the product to meet the needs of the procuring agency.

common issue area — A class of concern that is basic or common to almost all programs. PSM defines seven common software issues. See ***issue.***

component — Any separately-identifiable software or system element or unit. A component may be defined at any size or functional level within a product. A component may be defined as any of the structural elements that are commonly defined for products including units, design modules, or configuration items. A component may refer to any entity in the structure of a system. Since there is little agreement within the software communities on structural terminology, "component" may take on many different meanings. The supplier's project plan should clearly articulate the approach to creating the product structure and the definitions that will be used for the components. Components can be units, configuration items (CIs), objects, interfaces, screens, reports, packages, subsystems, icons, assemblies, or other measurable product structures. Problem reports and change requests are sometimes considered to be components, especially with respect to software maintenance activities during the operations and maintenance phase. COTS/GOTS and other non-developed parts or reusable software products can also be counted as components. Some components can be aggregated to form higher level components (for example, units to CIs to versions). These can be referred to as sub-components.

Configuration Item (CI) An aggregation of software and system products that satisfy an end-use function and is designated for separate configuration management by the acquirer. CIs are selected based on tradeoffs of various factors; including function, size, host or target computers, support concept, plans for reuse, criticality, interface considerations, the need to be separately documented and controlled, and other factors.

Cost/Schedule Control System Criteria (C/SCSC) DoD requirements that define what a contractor's management control system must have to qualify for bidding on selected military program acquisitions. The criteria include requirements for integrating cost, schedule, and technical performance measurements using the Work Breakdown Structure (WBS) and earned-value accounting methods. C/SCSC facilitates the analysis of variances from planned activities, and provides a means to estimate the cost of the contract at completion. This requirement is no longer required on new contracts as long as the contractor has an approved Earned Value Measurement System in place. See ***earned value.***

customer The individual or organization that procures software or system products for itself or another organization. See ***acquirer***.

cyclomatic complexity A measure of the logical complexity of a unit, based on the number of unique paths through the unit. This measure is used to evaluate code quality and to predict testing effort.

data item An attribute that is quantified to produce a measure. For example, the Number of Labor Hours is one data item.

defect A product's nonconformance to its specification; or any error in documentation, requirements, design, code, test plans, or any other work product. Defects are discovered during reviews, tests, and operations.

developer An organization that develops software and systems. The term "develop" may include development, modification, reuse, reengineering, operations and maintenance, or any other activity that results in software or systems. The developer may be a contractor, government agency, or internal development organization.

development The activities that result in software products, including requirements analysis, design, implementation, and integration and test. This term is used throughout PSM to describe the second of three phases in the software life cycle. See ***program planning, operations and maintenance.***

earned value The value of completed work, expressed in terms of the budget assigned to that work. Earned Value Measurement (EVM) is a cost management technique that relates resource planning to technical, cost, and schedule requirements. The earned value process budgets and schedules all work activities into time-phased increments that establish a cost and schedule measurement baseline.

estimation The type of analysis that is conducted to establish target values for software size, effort, and schedule to support project planning. Estimation usually starts with historical data and a set of assumptions about the project's process and products. Estimation not only produces estimates, but also identifies uncertainties that feed back into the issue identification process. Estimation should be conducted during the initial planning activity and during all subsequent plans.

estimator A special type of indicator used in estimation. An estimator describes a relationship between two quantities so that values of one can be used to estimate values of the other.

For example, the average productivity of an organization is an estimator that can be used with an appropriate measure of size to predict the amount of effort required for a project.

evaluate measurement One of the four basic measurement activities of the PSM process. The evaluation activity identifies potential improvements to the project's measures and measurement process. See ***apply measures, tailor measures, implement process***.

expected (planned) Planned or historical measurement data, such as milestone dates, target level of **value** reliability, or required productivity. See ***measured value.***

experience base A store or repository in which lessons learned and measurement artifacts are collected for use by later projects, or in later phases of the current project. The experience base may be kept in electronic or paper form.

failure A situation in which the system or system component does not perform a required function within specified limits.

feasibility analysis The type of analysis that is conducted to determine whether project plans and targets are technically realistic and achievable. Feasibility analysis uses historical data, experience, and consistency checks to evaluate the project plans. Any risk identified during this analysis should be entered into the project's risk management process. Feasibility analysis should be conducted during the initial planning activity and during all subsequent plans.

function point A software size measure that describes the level of information-processing functionality contained within a software product. Function points are derived early in the software life cycle from requirements or design specifications, and are applied across diverse application domains and technology platforms.

implement process In the PSM process, this term refers to one of the four basic measurement activities which comprise the software measurement process. The implement process activity includes all the steps to establish commitment and deploy a measurement process within an organization.

increment A functional subset of a system. Large systems are typically developed a series of increasingly complete increments (builds.)

indicator A measure or combination of measures that provides insight into a software issue or concept. PSM frequently uses indicators that are comparisons, such as planned versus actual measures. Indicators are generally presented as graphs or tables.

information system A combination of computer hardware and software, data, and telecommunications that performs functions such as collecting, processing, transmitting, and displaying information. Excluded are hardware and software computer resources that are physically part of, dedicated to, or essential to real-time mission performance of weapon systems.

issue A concern where obstacles to achieving program objectives might arise. Issues include risks, problems, and lack of information. These three types of issues are defined as:

problem: An ongoing or expected obstacle to achieving a project objective. While the magnitude and impact of a problem may not be known, it is assumed to be nearly certain to occur if not already occurring. Lack of qualified personnel may a known problem that has to be dealt with. Because the situation already exists, it isn't a risk.

risk: An obstacle that could occur, but is not certain. A risk is a potential problem. Risks represent the potential for the realization of unwanted, negative consequences from a project event. For example, a project plan may be based on the assumption that a COTS component will be available on a given date. There is a possibility (probability) that the COTS may be delayed and have some amount of negative impact on the project.

lack of information: A potential obstacle to the achievement of a goal arising from a lack of appropriate historical or contextual data on which to base plans or to evaluate performance. For example, an organization that has never developed C++ code has a lack of information about the level of productivity that it can achieve in C++. Even if no risk events occur, a C++ project might fail to achieve its objectives because those objectives were developed in a vacuum.

lack of information One of three categories of issues. See ***issue***.

life-cycle phase PSM defines three major life-cycle phases: project planning, development, and operations and maintenance. Four principal software activities occur within the development and operations and maintenance phases: requirements analysis, design, implementation, and integration and test.

low-level data Measurement data that is aggregated, collected, and reported at a level of detail that allows for the isolation of problems and for overall analysis flexibility. Aggregation of data is commonly at the activity level (requirements analysis, design, implementation, and integration and test), the component level, and the function level.

maintenance See ***operations and maintenance***.

measure A method of counting or otherwise quantifying characteristics of a process or product. Measures assign numerical values assigned to attributes according to defined criteria.

measured (actual) value Actual, current measurement data, such as hours of effort expended or lines of code produced. See ***expected value***.

measurement The process of assigning quantitative values to measures and indicators, according to some defined criteria. This process can be based on estimation or direct measurement. Estimation defines planned or expected measures. Direct measurement results in actual measures.

measurement analysis The critical examination of measures and indicators to identify problems, assess problem impact, project an outcome, or evaluate alternatives related to project issues. See ***estimation***, ***feasibility analysis***, and ***performance analysis***.

measurement analyst The person or team responsible for tailoring and applying measures for a given program or organization.

measurement category A set of related measures. Each common issue defined in PSM has one or more corresponding measurement categories. Software measures that provide the same type of information are grouped under the same measurement category. Each category answers different types of software-related questions.

measurement information Knowledge derived from analysis of measurement data and measurement indicators.

metric See measure, ***indicator***.

milestone A scheduled event for which some project member or manager is held accountable. A milestone is often used to measure progress.

normalization The process of transforming measures from different sources so that they can be combined or compared. For example, to compare the quality of work produced in two programs, it would be necessary to look at defect counts in relation to the amount or size of the product. This often requires defining and validating conversion rules and/or models.

objective A business goal or technical performance requirement allocated to a software or systems project. Objectives may include targets such as budgets, schedules, response times, and reliability.

operations and maintenance The activities necessary to ensure that an installed, operational system continues to perform as intended. This term is used throughout PSM to describe the third of three phases in the software life cycle. Software development can take place during the operations and maintenance phase. See ***project planning, development***.

performance analysis The type of analysis that is conducted to determine whether software development efforts are meeting defined plans, assumptions, and targets. Planned and actual performance data are the inputs to this process. The performance analysis process is designed to identify risks, problems, and corrective actions that can be taken. Performance analysis should be conducted periodically once a project has committed to a plan.

planned data See ***expected value***.

problem One of three categories of issues. See ***issue***.

problem report A documented description of a defect, unusual occurrence, observation, or failure that requires investigation, and may require product modifications.

project The people, processes, and organizations responsible for developing or supporting a system, either as a stand-alone system or as part of a larger system.

project manager The official responsible for acquiring, developing, or supporting a system to meet technical, cost, schedule, and quality requirements. Acquisition, development, and support includes both internal tasks and work that is contracted to another source.

project planning The activities necessary to define product requirements, assess and select suppliers, and develop project plans. This term is used throughout PSM to describe the first of three phases in the software life cycle. See ***development, operations and maintenance.***

repeatability The ability of two analysts to perform the same measurement analysis and to arrive at the same conclusions and recommendations.

rework Any effort to re-enact or correct work that has already been completed. Rework effort begins once a defect is found and continues until all of the work required to obtain acceptance of the reworked item is complete. Rework can also be measured in terms of size changes.

rippling Rippling occurs when a problem that arises in one issue area has an effect on another issue. For example, product size growth may cause effort overruns. Rippling multiplies the effect of an issue.

risk One of three categories of issues. See ***issue.***

Software Engineering Process Group (SEPG)	A group that facilitates the definition, maintenance, and improvement of the processes used by an organization.
technical manager	The person responsible for making decisions relating a specific discipline. For small projects this could be the project manager. Large projects often have software engineering, systems engineering, and hardware engineering managers.
supplier	An organization that enters into an agreement with the acquirer for the supply of a system, software product, or software service under the terms of that agreement.
structures	See ***aggregation structure***.
tailor measures	In the PSM process, this term refers to one of the four basic measurement activities which comprise the software measurement process. The tailoring activity includes identification and prioritization of program issues, selection and specification of appropriate software measures, and integration of the measurement requirements into the supplier's software process. See ***apply measures, implement process, evaluate measurement***.
traceability	The ability to link conclusions and recommendations to the project measures using a defined sequence of activities.
user	The agent or organization who will employ and operate a software or system product in the intended target environment.
weapon system	Items that can be used directly or indirectly by the armed forces to carry out combat missions.
Work Breakdown Structure (WBS)	A work breakdown structure defines the - elements associated with program work activities and products. Many measures are aggregated and analyzed at various WBS levels.

[This page intentionally left blank.]

Acronyms

A&T	Acquisition and Technology
ACWP	Actual Cost of Work Performed
AIS	Automated Information System
BCWP	Budgeted Cost of Work Performed
BCWS	Budgeted Cost of Work Scheduled
C/SCSC	Cost/Schedule Control System Criteria
C/SSR	Cost/Schedule Status Reports
C4I	Command, Control, Communications, Computers, and Intelligence
CMM	Capability Maturity Model
COCOMO II	Constructive Cost Model
COTS	Commercial Off The Shelf
CPR	Cost Performance Report
CI	Configuration Item
DAB	Defense Acquisition Board
DSMC	Defense Systems Management College
DT&E	Development, Test, and Evaluation
E&MD	Engineering and Manufacturing Development
EVMS	Earned Value Measurement System
GAO	General Accounting Office
GOTS	Government Off The Shelf
IFPUG	International Function Point Users Group
I-C-M	Issue-Category-Measure (Mapping)
IPPD	Integrated Product and Process Development
IPT	Integrated Project Team
ISO/IEC	International Organization for Standardization / International Electrotechnical Commission

ISSA	Inter-Service Support Agreement
IV & V	Independent Verification and Validation
JGSE	Joint Group on Systems Engineering
JLC	Joint Logistics Commanders
LAN	Local Area Network
LOC	Lines of Code
MAISAP	Major Automated Information System Acquisition Program
MAISRC	Major Automated Information System Review Council
MDAP	Major Defense Acquisition Program
MIS	Management Information System
MOA	Memorandum of Agreement
MOU	Memorandum of Understanding
NDI	Non-Developed Item
OSA	Open Systems Architecture
OSD	Office of the Secretary of Defense
OT&E	Operational Test and Evaluation
OUSD	Office of the Under Secretary of Defense
PSM	Practical Software and Systems Measurement
RFP	Request for Proposal
SEI	Software Engineering Institute
SEPG	Software Engineering Process Group
SISMA	Streamlined Integrated Software Metrics Approach
SPC	Software Productivity Consortium
STEP	Software Test and Evaluation Panel
SQA	Software Quality Assurance
WBS	Work Breakdown Structure

Bibliography

This bibliography lists measurement references that augment or support the guidance included in *Practical Software and Systems Measurement*. Readers may wish to consult these resources for additional information. Brief annotations are provided to describe each reference. The books are generally available through most technical publishers and bookstores. Government documents are available through the National Technical Information Service, 5285 Port Royal Road, Springfield, VA 22161, 703-487-4650.

Software Measurement References

Abts, Chris and Barry W. Boehm, A. Winsor Brown, Sunita Chulani, Bradford K. Clark, Ellis Horowitz, Ray Madachy, Donald J. Reifer, and Bert Steece, 2000, ***Software Cost Estimation with COCOMO II***, Prentice Hall PTR, Upper Saddle River, NJ.

This book describes an updated version of the COCOMO software cost estimation model first published in 1981. In addition to the describing the model and providing case studies of its use, there are descriptions of other emerging models that address the areas of quality, COTS software development, and rapid application development. The book includes a CDROM containing software tools, documents, and pointers to the COCOMO suite website.

Baumert, John H., and Mark S. McWhinney, September 1992, ***Software Measures and the Capability Maturity Model***, CMU/SEI-92-TR-25, ESC-TR-92-025, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

A reference that identifies which software measures can reasonably be expected at the various levels of SEI software process maturity. This book includes example graphs and advice on how to report specific measures.

Boehm, Barry W., 1981, ***Software Engineering Economics***, Englewood Cliffs NJ: Prentice-Hall.

This is a primary reference on how to use measurement and cost estimation to manage software projects and make business decisions based on quantitative tradeoffs. While the book's discussion of COCOMO has been updated with a new book, there are discussions on cost-effectiveness analysis, dealing with uncertainties, estimation procedures, and software lifecycle cost estimation.

Brooks, Frederick O., Jr., 1975, ***The Mythical Man Month: Essays on Software Engineering***, Reading, MA: Addison-Wesley Publishing Company.

This is a primary reference for software engineering. This book relates key lessons learned in managing a large software program and provides an overall perspective for the project manager.

Carleton, Anita D., Robert E. Park, Wolfhart B. Goethert, William A. Florac, Elizabeth K. Bailey, and Shari Lawrence Pfleeger, September 1992, ***Software Measurement for DoD Systems: Recommendations for Initial Core Measures***, CMU/SEI-92-TR-19, ESC-TR-92-019, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

This reference provides recommendations and a rationale for the SEI-defined Core Measures. The Core Measures include size, effort, schedule, and quality (measured in terms of defects and problem reports) and address issues common to almost all software programs.

Deming, W. Edwards, 1986, ***Out of the Crisis***, Cambridge, MA: Massachusetts Institute of Technology, Center for Advanced Engineering Study.

This book describes the quality crisis across a number of industries and relates effective strategies for dealing with them, focused on the use of Statistical Process Control techniques.

Dumke, Reiner R., 1993, ***Software Metrics: A Subdivided Bibliography***, Magdeburg, Germany: Technical University "Otto von Guricke" of Magdeburg.

This bibliography provides a comprehensive guide to both research and practical publications in software measurement, grouped by topic.

Fenton, Norman E., 1991, ***Software Metrics: A Rigorous Approach***, London: Chapman & Hall.

This book advocates a rigorous approach to software measurement that is based on fundamental measurement theory. It argues that much of modern software measurement is flawed because it ignores measurement fundamentals. This book gives the reader specific tools to overcome these deficiencies and put a measurement program on solid theoretical ground. This book is for the reader who desires a more theoretical treatment of software measurement than is found in PSM.

Florac, William A. and Anita D. Carleton, 1999, ***Measuring the Software Process: Statistical Process Control for Software Process Improvement***, Reading, MA: Addison Wesley.

This book is based on lessons and experiences in software process improvement initiatives and from implementing software measurement practices. It represents using measurement to control, improve, and predict, using statistical process control techniques.

Florac, William A., Park, Robert E., and Carleton, Anita D, April 1997, ***Practical Software and Systems Measurement: Measuring for Process Management and Improvement***, CMU/SEI-97-HB-003, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

This guide is based on the PSM process and principles of statistical process control. It identifies an effective approach for using performance data to manage and improve software processes.

Florac, William A., with the Quality Subgroup of the Software Metrics Definition Working Group and the Software Process Measurement Project Team, September 1992, ***Software Quality Measurement: A Framework for Counting Problems and Defects***, CMU/SEI-92-TR-22, ESC-TR-92-022, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

This reference provides a framework for counting problems and defects in software and using them to assess quality, which is one of the SEI Core Measures. It includes checklists that allow the reader to define how defects are actually defined and counted.

Goethert, Wolfhart B., Elizabeth K. Bailey, Mary B. Busby, with the Effort and Schedule Subgroup of the Software Metrics Definition Working Group and the Software Process Measurement Project Team, September 1992, ***Software Effort and Schedule Measurement: A Framework for Counting Staff-Hours and Reporting Schedule Information***, CMU/SEI-92-TR-21, ESC-TR-92-021, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

This reference provides frameworks for counting software staff-hours and schedule, both of which are SEI Core Measures. It includes checklists that allow the reader to define how staff-hours and schedule data are actually defined and counted.

Grady, Robert B., and Deborah L. Caswell, 1987, ***Software Metrics: Establishing a Company-Wide Program***, Englewood Cliffs, NJ: Prentice-Hall.

This book describes how Hewlett-Packard's corporate measurement program was implemented. It includes information on topics that range from how to compute specific measures to how to sell a measurement program to senior management.

Grady, Robert B., 1992, ***Practical Software Metrics for Project Management and Process Improvement***, Englewood Cliffs, NJ: Prentice-Hall, Inc.

This book examines more detailed issues with respect to software measurement, and more specifically relates measurement to software process improvement. It builds on information from the previous reference.

Hetzel, Bill, 1993, ***Making Software Measurement Work: Building an Effective Measurement Program***, Boston, MA: QED Publishing Group.

This book addresses how to get measurement implemented in an organization. It emphasizes fundamentals, explains how to begin, and includes a list of measurement tools and services available at the time of publication.

The Institute of Electrical and Electronics Engineers, Inc., 2001*,* ***IEEE Standard for Software Life Cycle Processes--Risk Management***, IEEE Std 1540-2001, New York, NY.

This IEEE Standards product is part of the family on Software Engineering. A process for the management of risk in the life cycle of software is defined. It can be added to the existing set of software life cycle processes defined by the IEEE/EIA 12207 series of standards, or it can be used independently.

International Function Points Users Group, revisions from 1994 to 2001, ***Function Points Counting Practices Manual***, Westerville, OH.

This industry-established standard defines the rules for counting function point. This can be obtained at http://www.ifpug.org/publications/manual.htm.

International Function Points Users Group, revisions from 1994 to 2001, ***Guidelines to Software Measurement***, Westerville, OH.

This guidebook introduces the basic concepts of software measurement. It describes how the measurement process fits into other software activities, and provides guidance on implementing a measurement program. It reviews product and process measures, discusses indicators, and examines ways to use measurement results. This can be obtained at http://www.ifpug.org/publications/guidelines.htm.

International Organization for Standardization and International Electrotechnical Commission,, 2000, ***Information Technology–Process Assessment,*** ISO/IEC 15504: 2000

This international standard provides a framework for the assessment of software processes.

International Organization for Standardization and International Electrotechnical Commission, 2002, ***Software Engineering –Software Measurement Process***, ISO/IEC 15939: 2002.

This International Standard defines a software measurement process applicable to all software-related engineering and management disciplines. The process is described through a model that defines the activities of the measurement process that are required to adequately specify what measurement information is required, how the measures and analysis results are to be applied, and how to determine if the analysis results are valid. The software measurement process is flexible, tailorable, and adaptable to the needs of different users.

International Organization for Standardization and International Electrotechnical Commission, 1998, ***Information Technology - Software Measurement -Definition of Functional Size Measurement***, ISO/IEC 14143: 1998.

Park, Robert E., with the Size Subgroup of the Software Metrics Definition Working Group and the Software Process Measurement Project Team, September 1992***, Software Size Measurement: A Framework for Counting Source Statements***, CMU/SEI-92-TR-20, ESC-TR-92-020, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

This reference provides a framework for counting source lines of code (SLOC) and using them to assess software size, which is one of the SEI Core Measures. It includes checklists that allow the reader to define how SLOC are actually defined and counted.

Paulk, Mark C., Bill Curtis, Mary Beth Chrissis, and Charles V. Weber, 1993***, Capability Maturity Model for Software***, Version 1.1, CMU/SEI-93-TR-24, ESC-TR-93-177, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

This reference describes a software process maturity framework that forms the basis for assessing the capability of a software organization. Five maturity levels and the key practices within each level are described.

Putnam, Lawrence H., and Ware Myers, 1996, ***Controlling Software Development***, IEEE Computer Society Executive Briefing. IEEE Computer Society Press.

This short booklet is aimed at executives and managers. It describes important concepts related to planning and tracking individual software projects, and also addresses how this information is used for longer-term software process improvement. Topics include estimating, defect management, measuring progress, productivity, and more. The booklet includes various sample indicators and explains how they should be used and interpreted.

Putnam, Lawrence H., and Ware Myers, 1992***, Measures for Excellence: Reliable Software on Time, within Budget***, Englewood Cliffs, NJ: Prentice-Hall.

This book focuses primarily on using tools for automated size estimation and project tracking, and also discusses life-cycle models, life-cycle management, and productivity analysis. It includes observations about patterns of software behavior, based on Putnam's historical database of software projects.

Rozum, J. A., and Iyer, S., 1996, ***A Data Definition Framework for Defining Software Measurements***, Technical Report, Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University.

Capability Maturity Model®-Integrated for Systems Engineering/Software Engineering, Version 1.0., CMU/SEI-2000-TR-018 (Staged Representation) and CMU/SEI-2000-TR-019 (Continuous Representation), Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, August 2000.

This International Standard defines a software measurement process applicable to all software-related engineering and management disciplines. The process is described through a model that defines the activities of the measurement process that are required to adequately specify what measurement information is required, how the measures and analysis results are to be applied, and how to determine if the analysis results are valid. The software measurement process is flexible, tailorable, and adaptable to the needs of different users.

Software Productivity Consortium, ***Software Measurement Guidebook***, SPC-91060-CMC, Version 02.01.00, August 1994, Software Productivity Consortium, Herndon, VA, and International Thompson Computer Press, 1995.

This reference provides detailed information that helps to define and interpret a software measurement process. It contains detailed guidance on a number of software measures. It provides a process maturity framework that supports assessing software and system engineering processes.

PSM Project Information

Practical Software and Systems Measurement Attribution

One of the primary purposes of *Practical Software and Systems Measurement: A Foundation for Objective Project Management* is to encourage the widespread implementation of software measurement throughout the DoD, government, and industry. The information included in the Guide was developed by a group of measurement professionals who gave much of their own time and effort to help meet this objective.

We encourage you to make direct use of the material contained in *Practical Software and Systems Measurement*. We ask that you acknowledge the source of the information as:

Practical Software and Systems Measurement: A Foundation for Objective Project Management, Version 4.0b, February 2003.

Additional copies of this Guide are available in electronic formats.

Project Contact Information

Practical Software and Systems Measurement: A Foundation for Objective Project Management is intended for those software acquisition and development organizations who need to more objectively plan, implement, control, and evaluate their software programs.

If you would like more information on using Practical Software and Systems Measurement, or on available PSM products and services please contact:

PSM Support Center
Cheryl Jones
US Army
TACOM-ARDEC
AMSTA-AR-QAT-A
Building 62
Picatinny Arsenal, NJ 07806-5000

(973) 724-5638 (Voice)
(973) 724-2382 (Fax)
880 (DSN)
psm@pica.army.mil
http://www.psmsc.com

[This page intentionally left blank.]

Practical Software and Systems Measurement Guide

Evaluation and Comment Form

We welcome any comments that will help us improve Practical Software and Systems Measurement. Please provide your inputs via hardcopy or email using the information provided below. If you email your comments, please write "PSM Guide (include version #) Evaluation" in the subject line.

PSM Support Center
Cheryl Jones
US Army
TACOM-ARDEC
AMSTA-AR-QAT-A
Building 62
Picatinny Arsenal, NJ 07806-5000
Phone: (973) 724-5638
Fax: (973) 724-2382
Email: psm@pica.army.mil

Name:
Date:
Organization:
Street Address:
Email address:
Telephone:
Fax:

Version of PSM Reviewed: _____

Place X before Part of Guide Commented On:	
Front Material:	Part 5, Indicator Examples
Part 1, The Measurement Process:	Part 6, Implement Process
Part 2, Tailor Measures:	Part 7, Evaluate Measurement
Part 3, Measurement Selection and Specification Tables:	Part 8, Project Measurement Case Studies
Part 4, Apply Measures:	Part 9, Supplemental Information

Would you like to receive updates to the Guide?

Overall Value: (Excellent, Good, Fair, Not Useful) **Please provide explanation.**

General Comments:

Specific Comments on Sections:

Section:	Page #:	Comments:

Use Additional sheets if needed.

Thank you.