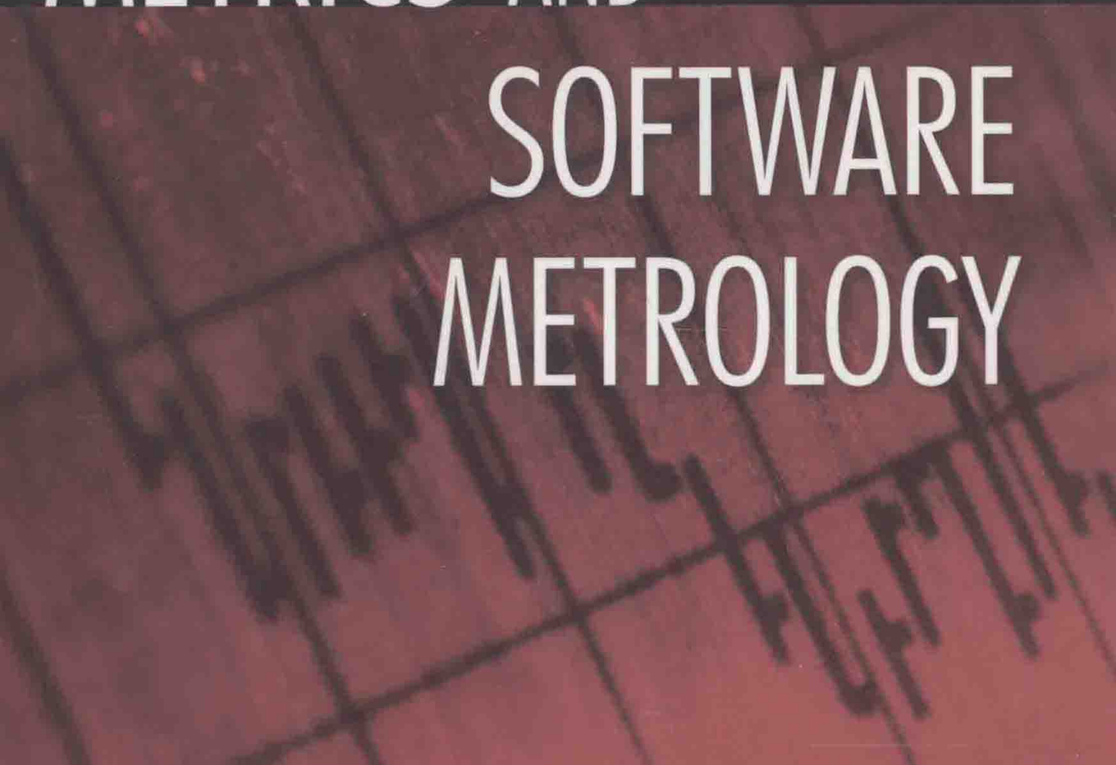
SOFTWARE METRICS AND SOFTWARE METROLOGY



ALAIN ABRAN

WILEY

IEEE

IEEE computer society

SOFTWARE METRICS AND SOFTWARE METROLOGY

Alain Abran

IEEE
computer
society

WILEY

A JOHN WILEY & SONS, INC., PUBLICATION

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.
Published simultaneously in Canada.

Limit of Liability/Disclaimer of Warranty: While the publisher and author have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives or written sales materials. The advice and strategies contained herein may not be suitable for your situation. You should consult with a professional where appropriate. Neither the publisher nor author shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

For general information on our other products and services or for technical support, please contact our Customer Care Department within the United States at (800) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic formats. For more information about Wiley products, visit our web site at www.wiley.com.

Library of Congress Cataloging-in-Publication Data is available.

ISBN: 978-0-470-59720-0

Printed in the United States of America.

10 9 8 7 6 5 4 3 2 1

SOFTWARE METRICS AND SOFTWARE METROLOGY

IEEE
computer
society

IEEE

FOREWORD

Software organizations must respond to increasingly demanding customers in a globally competitive market and must implement best industry practices. With services and products available from vendors the world over, customers are insisting that their software systems be of high quality and with support services that challenge those of the competition while costing as little as possible.

To satisfy these demands, software organizations must have the ability to develop and maintain software to meet the customer's needs, and it must have access to software that support the company's business processes.

- How do you know and how do you objectively demonstrate to your customers that your software organization is performing at the top of the industry?
- Can you leverage this knowledge to develop estimation skills as a competitive advantage?

Benchmarking and estimation is based on measurements. There is a tremendous need for software measures to support software performance measurement, benchmarking and software project estimation, even more so when software is contracted out to third party suppliers.

There is currently available a large number of software measures and quantitative models proposed to the practitioners' community for estimating software projects and measuring the quality of the software delivered. For instance, there are hundreds of measures proposed for software quality, software complexity, objects oriented as well as an impressive number of estimation models

But ...

- How many software organizations today have in place software measurement programs and use these measures and models as a basis for decision-making?

There must be then something at work that impairs the use of quantitative data for decision making in software-base organizations.

- What is it?

Within the software measurement community that has produced this large inventory of measures and quantitative models, there is a presumption that the lack of use of software measures in industry is caused by the practitioners' and managers' resistance to change.

This book is based on a different analysis and understanding of this lack of use of software measures by industry: this chasm comes from a lack of credibility in the practitioners communities, and this lack of credibility comes from the immaturity and unreliability of the measures themselves proposed to date to the industry.

Up until recently, software 'metrics' have been most often proposed as the quantitative tools of choice in software engineering, and the analysis of these had been most often discussed from the perspective referred to as 'measurement theory'.

However, in other disciplines, it is the domain of knowledge referred to as 'metrology' that is the foundation for the development and use of measurement instruments and measurement processes.

In this book, we use as a foundation the sets of measurement concepts documented in the ISO VIM (International Vocabulary of Basic and General Terms in Metrology) to document and compare the state of maturity of measures in software with respect to classic domains of science and engineering.

- This helps in particular to document practical aspects with respect to the current design of software measures and to identify the strengths and weaknesses of their own design as measures.

What was still missing is the ***know-how*** about how to correctly design software measures, and how to recognize if a software measure is well designed, and worth using as a basis for decision-making. This book focuses precisely on these two issues.

It is up to you:

- to acquire such ***know-how*** about the design of a software measure and
- to run with it for the benefit of your organization.

PREFACE

A book on the design of software measures must be suited to software engineers, both practitioners and researchers.

This book presents a perspective on software measurement that, on the one hand, is new in software engineering and, on the other hand, is fairly classical in most domains of sciences, engineering, and even in all areas of business.

Here, we share years of experience in the design of software measures for their successful use as decision making tools by software managers.

Because measurement is a fundamental engineering concept, software organizations of all sizes can use this book, and managers will find in it effective strategies for improving software management, along with numerous illustrative examples.

Applying the best practices in software measurement will ensure that software engineers and managers are equipped to respond to the most demanding customers, feel supported by senior executives and are proud to be part of the software team.

In addition, this book introduces many of the theoretical concepts and references needed by professionals, managers and students to help them understand the fundamentals of the identification and evaluation of software development and maintenance processes, and of improvements to them.

This book is intended for those developing, maintaining and managing software as well as for those in software process improvements.

STRUCTURE AND ORGANIZATION OF THIS BOOK

This book is organized into four (4) parts and fourteen (14) chapters.

Part 1: Key Concepts for the Design of Software Measures

A number of the software measures proposed to the industry have deficiencies severe enough to make some of them useless to practitioners. Part 1 presents in chapters one through five the key concepts in measurement that are necessary to recognize whether the design of a software measure is sufficiently strong to be meaningful in practice. Part 1 introduces, as well, the measurement terminology that is common in most fields of science and engineering; that is, of the metrology and related ISO standards on software measurement.

Chapter 1: Introduction.
This chapter presents the current level of maturity of software measurement within the software engineering discipline.

Chapter 2: From Measurement Methods to Quantitative Models: A Measurement Context Model.
This chapter presents a model to understand the key concepts of software measurement as well as the measurement terminology that is consistent with measurement in all disciplines. This chapter also discusses the process necessary to design a software measurement method.

Chapter 3: Metrology and Quality Criteria in Software Measurement.
This chapter presents the set of classical concepts in metrology, and presents various definitions and quality criteria in classical measurement.

Chapter 4: Quantification and Measurement are not the Same.
This chapter presents some of the differences between quantification and measurement, and establishes a parallel with the ISO 15939 Measurement Information Model.

Chapter 5: The Design of Software Measurement Methods.
This chapter presents the key concepts and steps required to design and evaluate software measurement methods, including defining the measurement principle in software measurement up to post-design activities.

Part 2: Some Popular Software Measures: How Good Are They?

Some software measures are currently popular in the industry, often because they are easy to collect or because they appear to take into account a large number of the practitioners concerns. However, in software measurement, being popular and widely quoted is not synonym to being good. Part 2 uses in chapters six through ten the criteria from Part 1 to illustrate some of the major weaknesses in the design of a few of the software measures that are either widely used or widely quoted in the software industry.

Chapter 6: Cyclomatic Complexity Number: Analysis of its Design
Chapter 7: Hasltead's Metrics: Analysis of their Designs
Chapter 8: Function Points: Analysis of their Design.
Chapter 9: Use Case Points: Analysis of their Design.
Chapter 10: ISO 9126: Analysis of its Quality Models and Measures.

Part 3: The Design of COSMIC—ISO 10761

Part 3 illustrates in chapters eleven and twelve how the lessons learned from the analysis of the key concepts for the design of a software measure have been put into practice to design a software measurement method conformant to the ISO criteria for a measurement method of the functional size of the software, that is the COSMIC—ISO 19761. Part 3 focuses on the design process rather than on the details of this specific measurement method.

Chapter 11: COSMIC: Design of an Initial Prototype.

This chapter illustrates how this software measure of the functional size of software for real-time and embedded software was designed in response to an industry need. It describes in particular the process used to design the initial prototype of COSMIC, its field trials and its initial deployment.

Chapter 12: COSMIC—Scaling up and Industralization.

This chapter illustrates the additional effort to scale up COSMIC to increase its international acceptance and to bring it to be adopted as an international standard: ISO 19761. The key concepts of the COSMIC measurement method are also presented in this chapter.

Part 4: Other Issues in the Design of Software Measures

Part 4 illustrates in chapters thirteen and fourteen some additional issues that are traditional in measurement in day-to-day life, but that have not yet been seriously addressed in software measurement. Two specific examples are presented: convertibility across measurement design and measurement standard etalons.

Chapter 13: Convertibility across Measurement Methods

While numerous software measures are proposed for the same attributes, there is a scarcity of convertibility studies across alternative ways of measuring. This chapter presents a convertibility analysis across two functional size measurement methods: IFPUG Function Points and COSMIC Function Points.

Chapter 14: Design of Standard Etalons: The Next Frontier in Software Measurement

While measurement in science relies on well established standard etalons (such as for the meter and kilograms) to ensure the correctness and consistency of measurement results across contexts and countries, not a single standard etalon has yet been established for measuring software. This chapter looks at this next frontier in software measurement and reports on an initial attempt to design a first draft of a standard etalon for a referenced set of software requirements.

This book also contains three appendices:

Appendix A: List of Acronyms
Appendix B: Glossary of Terms in Software Measurement.
Appendix C: References

Additional material to complement the information contained in this book can be found at http://profs.logti.etsmtl.ca/aabran

If you are a software manager, you should:

Part of the book	Read chapters:	Why?	What to do with it?
1	1 & 2	Not all software measures have strong designs. These two chapters explain the quality criteria that should be expected from software measures.	Quality criteria your staff should look for when analyzing a software measure as a basis for decision making.
	3	Chapter 3 on Metrology: through a quick reading of this chapter, managers will get an understanding of why software measures are still far way from the maturity of other measures in science and engineering.	For information only.
	4	This chapter positions the key ISO standard on software measurement—ISO 15939—and clarifies the differences between measurement and quantitative models	For a better understanding of the subtleties in measurement and quantitative analysis
2	**One** of chapters 6 to 10	Select the chapter to read on the basis of which one of these measures are in use in your organization	For the measures in use in your organization, ask your software engineer to document the impact of the weaknesses identified in this book on the quality of the measures and models you use for taking decisions.

Part of the book	Read chapters:	Why?	What to do with it?
3	Chapter 12	For an example of a software measurement method that has been accepted as an ISO standard.	If your organization is using software measures that have not yet been standardized, you must be careful.
4	Chapter 13	To get an example of convertibility across two distinct measures of software.	If your organization is mixing together numbers from different measurement designs, you should get your software engineers to document the impact this has on your quantitative models for decision making.

If you are a software engineering practitioner or a software quality analyst *using or planning to use existing software measures* you should:

Part of the book	Read chapters:	Why?	What to do with it?
1	1 to 5	These chapters explain the quality criteria that should be expected from software measures. These chapters teach you what to look for when analyzing a software measurement method.	You should document the impact of the weaknesses identified in this book on the quality of the measures and models used in your organization for making decisions.
2	**One** of the chapters 6 to 10	For examples of major weaknesses in some of the popular measures proposed to the software industry.	If you are using one of these measures, you should review your actual interpretation and usage of the numbers you are getting out. If you are using other software measures, you should ask for similar analyses.
3	Chapter 12	Same as for managers	Same as for managers
4	Chapter 13	Same as for managers	Same as for managers

If you are in software process improvement or a researcher *planning to analyze existing software measures or to design new software measures* or **if you are taking an undergraduate or graduate course on software measurement** you should:

Part of the book	Read chapters:	Why?	What to do with it?
1	All chapters from 1 to 4	Same as in previous tables above and an in-depth discussion on the design of software measurement methods in Chapter 5.	Same as in previous tables. Use the design process described in chapters 2, 4 and 5, when you have to design a new measurement method for software or when you have to correct the design of an existing one.
2	All chapters from 5 to 10	For a good coverage of the analyses documented to date on the design of some of the popular measurement methods.	If you have to evaluate other software measures, these chapters will guide you on how to do such evaluations. If you have to design new software measures, these chapters will also teach you what mistakes to avoid.
3	Both chapters 11 and 12	For understanding the stages of developing a software measurement method, from its initial design up to its highest recognition in the field of measurement that is: an International Standard.	Use a process similar to the one in Chapter 11 for *introducing strengths* into the design of *new* measurement methods for software. Use a process similar to the one in Chapter 12 for *adding strengths* into the design of *existing* measurement methods for software.
4	Both chapters 13 and 14	For an understanding of two critical issues in software measurement: using distinct measurement methods for the same attributes and the lack of standard etalon in software measurement.	These issues still have to be tackled by researchers for the majority of measures proposed to practitioners.
	And All Advanced Readings sections	For an in-depth understanding of the issues presented in each chapter.	For detailed examples of concepts introduced in the various chapters.

This book is not about:

- A compendium of all software measures:
    - The purpose of this book is not to present an exhaustive list of measures of any type, or of a specific type (for instance on OO metrics).
    - There exists already on the market a number of books presenting inventories of alternative measures, as well as hundreds of research papers on emerging designs, which at this stage would still be fairly immature.
- A compendium of software estimation models:
    - This book does not list or discuss any of the estimation models for software.
    - For instance, COCOMO [Boehm 1981, 2000] is an 'estimation model' which attempts to predict the relationships across a large number of factors. COCOMO is not about measurement but a lot more about experimentation (as in science) to build prediction models. COCOMO, for instance, should be used and evaluated as an estimation model. This will be discussed in another book looking into the design and evaluation of estimation models.
- A compendium of analyses of all software measures:
    - This book presents from chapters six through ten analyses that have already been carried out in research and published at a number of international conferences.
    - A large number of software metrics, such as the ones in (or derived from) Chidamber & Kemerer metrics suite [Chidamber 1993], has not yet been analyzed from a metrology perspective. The analysis from a metrology perspective of these other measures still has to be done.

COSMIC Function Points

The COSMIC Function Points have been adopted in 2003 as an international standard—ISO 19761—for measuring the functional size of software. Having been designed to meet metrology criteria, COSMIC Function Points are at times used in this book to illustrate a number of measurement concepts. For more details on the design of COSMIC Function Points, see Section 5 of Chapter 12.

ACKNOWLEDGMENTS

A number of collaborators, including colleagues in industry and university as well as PhD students, have helped me over the years improve my understanding of many of the concepts presented in this book, in particular:

Chapter	Co-Contributor
2: From Measurement Methods to Quantitative Models: A Measurement Context Model	Dr. Jean-Philippe Jacquet (France)
3: Metrology and Quality Criteria in Software Measurement	Dr. Asma Sellami—University of Sfax (Tunisia)
4: Quantification and Measurement are not the Same.	Dr. Jean-Marc Desharnais—Ecole de technologie supérieure (Canada) & Bogaziçi University (Turkey)
5: The Design of a Software Measurement Method	Dr. Naji Habra—Facultés Universitaires Notre-Dame de la Paix—FUNDP, Namur (Belgium)
6: Cyclomatic Complexity Number: Analysis of its Design	Dr. Naji Habra—FUNDP (Belgium)
7: Halstead's Metrics: Analysis of their Designs	Dr. Rafa Al-Qutaish—Alain University of Science and Technology, Abu Dhabi Campus, United Arab Emirates
8: Function Points: Analysis of their Design	
9: Use Case Points: Analysis of their design	Joost Ouwerkerk—Expedia (Canada)
10: ISO 9126: Analysis of its Quality Models and Measures	Dr. Rafa Al-Qutaish—Alain University of Science and Technology, Abu Dhabi Campus, United Arab Emirates
11: COSMIC—Design of an Initial Prototype	D. St-Pierre, Dr. Desharnais, Dr. P. Bourque and M. Maya (École de technologie supérieure—University of Québec—Canada)

Chapter	Co-Contributor
12: COSMIC—Scaling up and Industralization	C. Symons, M. O'Neil, P. Fagg, and a number of the COSMIC members of the measurement practices committee
13: Convertibility Across Measurement Methods	Dr Desharnais—Ecole de technologie supérieure (Canada) & Bogaziçi University Turkey
14: Design of Standards Etalons: The Next Frontier in Software Measurement	Dr. Adel Khelifi—University Al Hosn (United Arab Emirates)

Above all, this book is dedicated to all those who provided me with feedback and insights on software measures over the years and who are contributing, each in his or her own way, to the improvement of software measures as a foundation for sound, quantitatively-based decision making.

ABOUT THE AUTHOR

Dr. Alain Abran is a professor and the director of the research group in Software Engineering Management at the École de Technologie Supérieure (ETS)—Université du Québec, Montréal, Canada (www.gelog.etsmtl.ca)

He is a co-editor of the *Guide to the Software Engineering Body of Knowledge* (www.swebok.org). He is actively involved with software engineering standards with ISO/IEC JTC1 SC7—Software and System Engineering—and has been its international secretary in 2001–2003. He is chairman of the Common Software Measurement International Consortium (COSMIC).

Dr. Abran has more than 20 years of industry experience in information systems development and software engineering and 15 years of university teaching. He holds a PhD in electrical and computer engineering (1994) from the École Polytechnique de Montréal (Canada) and Master's degrees in management sciences (1974) and electrical engineering (1975) from the University of Ottawa (Canada).

His research interests include software productivity and estimation models, software engineering foundations, software quality, software measurement, functional size measurement methods, software risk management and software maintenance management.

Most of his publications can be downloaded from: http://profs.logti.etsmtl.ca/aabran/Publications/index.html