A Roadmap to Maturity for Software Measures

Alain Abran
List of topics

1. Introduction
2. Metrology Concepts
3. A Measurement Process Model
4. A Measurement Body of Knowledge
5. Discussion
Introduction

 Software is an intellectual product
 Software is new

Widely held beliefs:

– We have to ‘invent’ how to measure software
– Software measurement is so unique that there is:
  – Not much in common from measurement of physical objects
  – Not much to learn from other fields of sciences
Introduction

When we measure physical objects, what do we measure?

– Objects

Or

– ….
Introduction

- What measurement infrastructure has been put in place at the national and international levels?
  - .....
Introduction

Any profession dedicated to measurement?

— ..... 
— .... 
— .... 
— .... 
— ....
Introduction

The dominant approach in software measurement:

- The ‘software metrics’ approach
  - Intuitive approach to the design of ‘metrics’
  - Large variety of individual proposals
  - Focus on ‘measurement theory’
  - Representation conditions
  - Mathematical properties
Introduction

Consequences of the dominant approach

Direct:
- Practitioners are not keen on using ‘software metrics’
- Experts disagree on the relevance of using ‘software metrics’: eg. Work on fundamental principles

Indirect:
- Limited design expertise
- Incomplete validation framework
- Weaknesses of models based on ‘unsound metrics’
List of topics

1. Introduction

2. Metrology Concepts

3. A Measurement Process Model

4. A Measurement Body of Knowledge

5. Discussion
Figure 2: Measurement foundations [ABRA02a]
Metrology Concepts

Figure 1: Model of the categories of metrology terms [ABRA02a]
Figure 3: Measurement Procedure [ABRA02a]
Classification of terms in the category of 'Measurement Results' [ABRA02a]

<table>
<thead>
<tr>
<th>Types of measurement results</th>
<th>Modes of verification of measurement results</th>
<th>Uncertainty of measurement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indication (of a measuring instrument)</td>
<td>Accuracy of measurement</td>
<td>Experimental standard deviation</td>
</tr>
<tr>
<td>Uncorrected result</td>
<td>Repeatability (of results of measurements)</td>
<td>Error (of measurement)</td>
</tr>
<tr>
<td>Corrected result</td>
<td>Reproducibility (of results of measurements)</td>
<td>Deviation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Relative error</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Random error</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Systematic error</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Correction</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Correction factor</td>
</tr>
</tbody>
</table>
Functional Size

A unique set of measures in software engineering:

- Designed in the late 1970’s:
  - By Albrecht, from IBM, using 24 MIS projects
- Published in the early 1980’s
- User group in the mid 1980’s
  - Measurement Manual
  - Training & Certification
Functional Size

Innovation = Standardization through ISO

A meta-standard to layout the ground rules about functional size measurement: ISO 14143

- Part 1 = Definitions of Key Concepts
- Part 2 = Conformity Assessment
- Part 3 = Verification Guide
- Part 4 = Set of References
- Part 5 = Functional Domains
- Part 6 = A Guide
Four specific methods approved by ISO

- ISO 19761: COSMIC-FFP
- ISO 20926: IFPUG
- ISO 20968: MKII
- ISO 24570: NESMA

- Will they withstand the test of time as measurement methods?
- Are there good measuring instruments?
- Are these instruments calibrated and certified?
ISO 9126 on Software Products Quality

- Part 1: Quality Models and Definitions
- Parts 2 to 4: +120 Metrics!
  - And little about:
    - measurement method for each of the +120 metrics
    - quality of measurement results.
  - Then (if used in a non-consistent manner), how do you figure out how measurement results compare across contexts, across time, and across measurers?
  - How do you benchmark?
List of topics

1. Introduction
2. Metrology Concepts
3. A Measurement Process Model
4. A Measurement Body of Knowledge
5. Discussion
High-level measurement process model

Step 1: Design of the measurement method

Step 2: Application of measurement method rules

Step 3: Measurement result analysis

Step 4: Application of measurement method rules

Source: Abran and Jacquet
Step 1  
**Design of the measurement method**

- Design of the objectives
- Characterization of the concept to be measured
- Definition of the numerical assignment rules
- Design or selection of the meta-model

Step 2  
**Measurement method application**

- Software documentation gathering
- Construction of the software model
- Application of the numerical assignment rules

Step 3  
**Measurement result analysis**

- Result

Step 4  
**Exploitation of the result (examples)**

- Quality model
- Budgeting model
- Productivity model
- Estimation model
Alignment of metrology concepts with the measurement process model

<table>
<thead>
<tr>
<th>Measurement process model</th>
<th>Design of measurement methods</th>
<th>Application of measurement method rules</th>
<th>Measurement results analysis</th>
<th>Exploitation of measurement results</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISO metrology model</td>
<td>Quantities and units</td>
<td>Measuring instruments</td>
<td>Measurement results</td>
<td></td>
</tr>
</tbody>
</table>
List of topics

1. Introduction
2. Metrology Concepts
3. A Measurement Process Model
4. A Measurement Body of Knowledge
5. Discussion
What is Software Engineering?

IEEE 610.12:

– “(1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

– (2) The study of approaches as in (1).”
Categories of Knowledge in the SWEBOK

- Generally Accepted
- Advanced and Research

Focus of the SWEBOK Guide
Generally Accepted

«Applies to most projects, most of the time, and widespread consensus validates its value and effectiveness»

Project Management Institute - PMI

Bachelor + 4 years of experience
Guide to the Software Engineering Body of Knowledge
(Version 0.95)

- Software Requirements
  - Requirement Engineering Process
  - Requirements Elicitation
  - Requirement Analysis
  - Requirements Specification
  - Requirements Validation
  - Requirements Management

- Software Design
  - Software Design Basic Concepts
  - Key Issues in Software Design
  - Software Structure and Architecture
  - Software Design Quality Analysis and Evaluation
  - Software Design Notations
  - Software Design Strategies and Methods

- Software Construction
  - Reduction in Complexity
  - Anticipation of Diversity
  - Structuring for Validation
  - Use of External Standards

- Software Testing
  - Testing Basic Concepts and Definitions
  - Test Levels
  - Test Techniques
  - Test-Related Measures
  - Managing the Test Process

- Software Maintenance
  - Basic Concepts
  - Maintenance Process
  - Key Issues in Software Maintenance
  - Techniques for Maintenance

SURA 2006, Cadiz (Spain), Nov. 6-8, 2006
Zelkowitz & Wallace
taxonomy of empirical
support methods

<table>
<thead>
<tr>
<th>Category / Empirical support method</th>
<th>Description</th>
<th>Weaknesses</th>
<th>Strengths</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A. Observational</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A1. Project Monitoring</td>
<td>Collect development data</td>
<td>No specific goals</td>
<td>Provides baseline for the future; inexpensive</td>
</tr>
<tr>
<td>A2. Case Study</td>
<td>Monitor project in depth</td>
<td>Poor controls for later replication</td>
<td>Can constrain one factor at low cost</td>
</tr>
<tr>
<td>A3. Assertion</td>
<td>Use ad-hoc validation technique</td>
<td>Insufficient validation</td>
<td>Serves as a basis for future experiments</td>
</tr>
<tr>
<td>A4. Field Study</td>
<td>Monitor multiple projects</td>
<td>Treatments differ across projects</td>
<td>Inexpensive form of replication</td>
</tr>
<tr>
<td><strong>B. Historical</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B1. Literature Search</td>
<td>Examine previous published studies</td>
<td>Selection bias; treatments differ</td>
<td>Large available database; inexpensive</td>
</tr>
<tr>
<td>B2. Legacy</td>
<td>Examine data from completed projects</td>
<td>Cannot constrain factors; data limited</td>
<td>Combines multiple studies; inexpensive</td>
</tr>
<tr>
<td>B3. Lessons Learned</td>
<td>Examine qualitative data from completed projects</td>
<td>No quantitative data; cannot constrain factors</td>
<td>Determine trends; inexpensive</td>
</tr>
<tr>
<td>B4. Static Analysis</td>
<td>Examine structure of developed product</td>
<td>Not related to development method</td>
<td>Can be automated; applies to tools</td>
</tr>
<tr>
<td><strong>C. Controlled</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C1. Replicated</td>
<td>Develop multiple versions of product</td>
<td>Very expensive; Hawthorne effect</td>
<td>Can control factors for all treatments</td>
</tr>
<tr>
<td>C2. Synthetic</td>
<td>Replicate one factor in lab setting</td>
<td>Scaling up; interactions among multiple factors</td>
<td>Can control individual factors; moderate cost</td>
</tr>
<tr>
<td>C3. Dynamic Analysis</td>
<td>Execute developed product for performance</td>
<td>Not related to development method</td>
<td>Can be automated; applies to tools</td>
</tr>
<tr>
<td>C4. Simulation</td>
<td>Execute product with artificial data</td>
<td>Data may not represent reality; Not related to development method</td>
<td>Can be automated; applies to tools; evaluation in safe environment</td>
</tr>
</tbody>
</table>
List of topics

1. Introduction
2. Metrology Concepts
3. A Measurement Process Model
4. A Measurement Body of Knowledge
5. Discussion
Discussion

Key challenge for the designers of software measures:

- Innovation or consensus building?
  - Promoting:
    - our ‘own new metrics’ or
    - robustness in terms of metrology related properties?
- How to figure out the key design aspects out of a bunch of alternative ‘metrics’ designs?
  - How to get to a consensus?
Discussion

- How do we build an infrastructure for software measures?

- What is the process to define an ‘étalon’ for a software measurement standard?
  - What are the design issues?
  - How do we tackled them?

- How to set up an ‘étalon’ for a specific software measure?
  - And how do we make it evolve?
The roadmap to software maturity?

- We must ensure that the fundamentals are right.
- We have to build upon centuries of knowhow on how to build measures.
- We have to contribute to the building of a software measurement infrastructure.
Thank You!

alain.abran@etsmtl.ca  www.gelog.etsmtl.ca