In search of software engineering principles:

1996-1998 Delphi studies to develop a group consensus

Robert Dupuis - Pierre Bourque
Alain Abran - James W. Moore
List of topics

• Introduction: Standards & Principles
• Delphi Studies – Objectives & Rounds
• Criteria and Participants
• Outcomes
Standards Strive to Balance Principles and Practice

Standards strive to integrate and organize strengths of *a priori* principles with ‘best’ practices observed in the messy real-world.

In many disciplines, *a priori* considerations are provided by science and mathematics. Sometimes they are provided by ‘traditions’ or by market forces. In software engineering, there is no agreement on such *a priori* and we have to discover and figure out what are its principles.
We have to invent 'what'?
LOG, 20/08/2007
Software Engineering Standards

Context

SWE Body of Knowledge

Fundamental Principles

Motivate

Education
Certification
Practice

Process Definition
Process Assessment
Process Measurement
Best Practices

Integrating Framework

Umbrella Standard(s)

IEEE
ISO/IEC
DoD

Integrated Collection of Practice Standards

IEEE
ISO/IEC

Quality Management
Project Management
System Engineering
Competency Certification
Organizational Adoption
Buyer/Seller Agreement
Critical System Assurance

Engineering Foundations of Software Engineering
ICEE Workshop – Coimbra Sept 2, 2007
Fundamental Principles of Software Engineering

A collaborative Effort: IEEE Computer Society & Université du Québec (UQAM-ETS)

SWE Principles are specific cases of general engineering principles.

SWE Principles organize, explain and validate the practice standards.

Some SWE Principles may be generalized to principles for the engineering of complex systems.

SWE Principles should be “abstractions” of practice standards.

Practice standards should be recordings of observed best practices.

Source: [Jabir97]
The 1996-1998 Search Process

1996 Decision of the IEEE Software Engineering Standards Committee

- Recommendation to identify fundamental principles of software engineering

Workshop - SES'96 (Montréal, October 1996)

- Criteria for identifying and evaluating proposed principles

Delphi I: International Software Engineering Experts

- Round 1:
  - Submission of 65 proposed principles
  - Synthesis into 16 candidates
  - 16 candidate fundamental principles

- Round 2:
  - Vote on the importance of each candidate
  - Mean rating for each candidate

- Round 3:
  - Concurrence with mean rating

Evaluated candidate fundamental principles by international experts

Workshop - ISESS'97 (Walnut Creek, CA, June 1997)

- Improved list of fundamental principles with recommendations for future steps

Delphi II: IEEE Computer Society Software Engineering Experts

- Round 1:
  - Vote on the importance of each candidate
  - Median rating for each candidate

- Round 2:
  - Concurrence with median rating

Candidate fundamental principles evaluated by IEEE CS experts

Web-Based Survey: CS-TCSE Members

- Survey within membership of the IEEE/TCSE

Candidate fundamental principles evaluated by practitioners

Steps in which comments are obtained

Engineering Foundations of Software Engineering
ICEE Workshop – Coimbra Sept 2, 2007
Criteria: Principles must be ...

- Less specific than methodologies
- More durable than methodologies and techniques
- Extracted from practice
- Linked to at least one underlying concept of SE
- Not involve a trade-off
- Be specific enough to be able to demonstrate experimentally that not applying the principle leads to bad consequences (e.g. undesirable outcomes).
Est-ce le bon mot en anglais?

LOG, 20/08/2007
Objectives of Delphi I (Three rounds)

• Evaluation of criteria
• Identify and evaluate candidate principles
• Propose recommendations on criteria & principles
Participants

- M. Azuma, Waseda University, Japan
- F.P. Brooks, U. of North Carolina, USA
- R.N. Charette, ITHABI Corp., USA
- P. DeGrace, Consultant, USA
- C. Ghezzi, Politecnico di Milano, Italie
- T. Gilb, Result Planning Ltd, Norway
- B. Littlewood, City University, G-B
Participants

• S. MacDonell, U. of Otago, New-Zealand
• T. Matsubara, Matsubara Consulting, Japan
• J. Musa, Consultant, USA
• R. Pressman, R.S. Pressman & Associates, USA
• M. Shaw, Carnegie-Mellon U. USA
• Two participants chose to remain anonymous
ROUND 1

• Objectives
  – Get candidate principles
  – Get justifications
  – «Formulate what you would consider five Fundamental Principles of Software Engineering»

• Results
  – 13 participants contributed
  – 65 suggestions

• Outcome
  – Consolidated into 16 PFs
    (Including elimination of duplicates)
ROUND 2

• Objectives
  – Evaluate (on a scale 1-10) the 16 candidates
  – Get additional justifications on the score

• Results
  – 10 participants contributed
ROUND 3

• Objectives
  – Measure the level of consensus on the average scores from Round 2 outcome
  – 12 participants contributed
Fundamental Principles of SE

A. Apply and use quantitative measurements in decision-making
B. Build with and for reuse
C. Control complexity with multiple perspectives and multiple levels of abstraction
D. Define software artifacts rigorously
E. Establish a software process that provides flexibility
F. Implement a disciplined approach and improve it continuously
G. Invest in the understanding of the problem
H. Manage quality throughout the life cycle as formally as possible
I. Minimize software component interaction
J. Produce software in a stepwise fashion
K. Set quality objectives for each deliverable product
L. Since change is inherent to software, plan for it and manage it
M. Since tradeoffs are inherent to software engineering, make them explicit and document them
N. To improve design, study previous solutions to similar problems
O. Uncertainty is unavoidable in software engineering. Identify and manage it
Recommandations

• Validate the list through a second Delphi study with experts of the IEEE software engineering community
  – [Walnut Creek, CA, June 1997]

• Cross-check with practitioners
  – members of the IEEE Computer Society
Recommendations

• Improve the list by using it to analyze:
  – the current SE standards portfolio [Walnut Creek, CA, June 1997]
  – the SE body of knowledge as stated in current textbooks
  – the SE curriculum
Modifications for Delphi II

• Removal of three candidates
• Addition of:
  – To improve design, study previous solutions to similar problems
  – Control complexity with multiple perspectives and multiple levels of abstraction
Delphi II

- 72 experts of the Computer Society:
  - *Technical Council on Software Engineering*
  - Editorial committees:
    - ‘*Software*’
    - ‘*Transactions on Software Engineering*’
- 30 participants contributed
- Results very similar to Delphi I
Last step: Web-based Survey of Practitioners

- Participants from 48 countries
- Employers: R&D, software development, education, ...
- Education level:
  - 43%: Ph.D.,
  - 35%: Masters,
  - 19%: Undergraduates
Last step: Web-based Survey

• # of years of experience in industry: :
  – 9% : 30+ years
  – 30% : 20-29 years
  – 48% : 10-19 years
  – 13% : 1-9 years
Data

• Quantitative:
  – Scores on each of the 15 candidate principles
  – Measures of consensus level relative to each score

• Qualitative:
  – Comments on candidate principles
Quantitative data analysis

• Average and median scores for each principle
• Assessment of consensus level on each principle
  – including standard deviation
Most popular

L. Since change is inherent to software, plan for it and manage it. (9.1-12)

G. Invest in the understanding of the problem. (8.7-10)

M. Since, trade-offs are inherent to software engineering, make them explicit and document them. (8.4-11)

P. Uncertainty is unavoidable in software engineering. Identify and manage it. (8.0-11)
Least popular...

N. The requirements must be firm and fixed (3.3 - 9)

O. The tools, methods, and support systems must be designed and selected to support the software engineers. (4.2 - 10)

C. Deal with different individual aspects of the problems by concentrating on each separately. (4.9 - 6)
Conclusions

• Delphi method proved appropriate when expert opinion is the main source of information

• Results from Delphi study can be used for surveys or experimentation
Some surprises...

A. Apply and use quantitative measurements in decision-making. (7.6 - 9)

F. Implement a disciplined approach and improve it continuously. (6.9 - 7)

D. Define software artifacts rigorously. (6.4 - 8)

H. Manage quality throughout the life cycle as formally as possible. (7.8 - 9)
Others...

B. Build with and for reuse. (8,0 - 9)
E. Establish a software process that provides flexibility. (7,6 - 10)
I. Minimize software components interaction. (7,3 - 11)
J. Produce software in a stepwise fashion. (7,7 - 8)
K. Set quality objectives for each deliverable product. (7,7 - 11)
Importance of ‘Measurement’ in SE: quantitative data analysis

• Selection of comments relative to measurement
• Issues identified from analysis of comments
• Links to literature
• Statement of research issues
Importance of ‘Measurement’ in SE: Analysis of qualitative data

Comments:

• « Measurement is important, but it should deliver value commensurate with the cost of collecting and analyzing information »

• « While I am a proponent of quantitative measurement, decisions should be based on the amount of information that is cost-effective to acquire given the importance ($) of the decision »
Importance of ‘Measurement’ in SE: Analysis of qualitative data

• *Issue*: Value attributed to measurement may not be higher than its costs

• *Link to literature*: How much effort should one invest in software defects measurement? (Weller 1994)

• *Question*: Is there a method to evaluate costs/benefits of measurement?