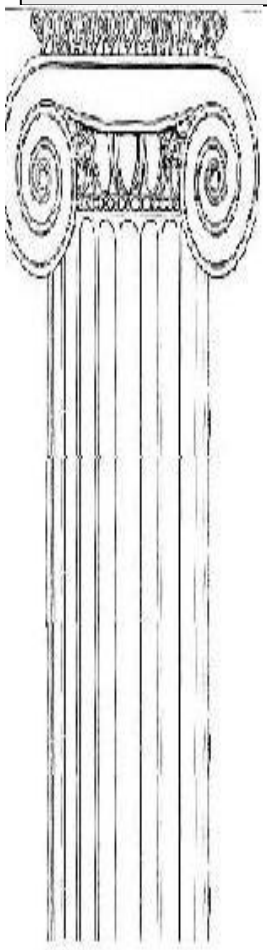
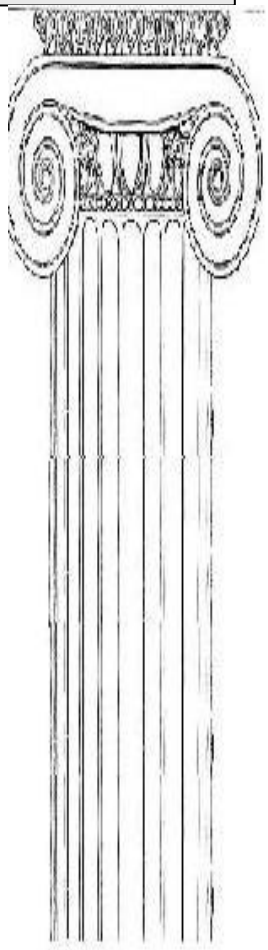
Software Engineering Principles: Do they Meet Engineering Criteria?

Kenza Meridji

Supervision : Dr. Alain Abran





Agenda

- Introduction
- Related work
- Methodology
- Discussion

Introduction

Software engineering is defined by the IEEE as:

"The application of a systematic, disciplined, quantitative approach to the development, operation and maintenance of software, the application of engineering to software'

IEEE 610.12

Related work

Individual work

Boehm, B.W., Seven Basics Principles of Software Engineering. Journal of Systems and Software, 1983. 3(no 1): p. 366-371.

Davis, A.M., 201 Principles of Software Development. 1995, New-York: McGraw-Hill.

Karl E. Wiegers Creating a software Engineering culture. 1996. New-York: Dorset House Publishing.

Séguin, N., Inventaire, Analyse et Consolidation des Principes Fondamentaux du Genie Logiciel. 2006, Université du Québec à Montréal: Montreal.

Collaborative work

Buschmann, F., R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, Pattern Oriented Software Architecture. England. Wiley ed. 1996, England. 476 pages.

Pierre Bourque, Robert Dupuis, Alain Abran, James W. Moore, Leonard Tripp, and S. Wolff, Fundamental principles of software engineering – a journey. Journal of Systems and Software, 2002. 62: p. 59-70.

Ghezzi, C., M. Jazayeri, and D. Mandrioli, Fundamentals of Software Engineering. 2th ed . 2003, New-Jersey: Prentice Hall.

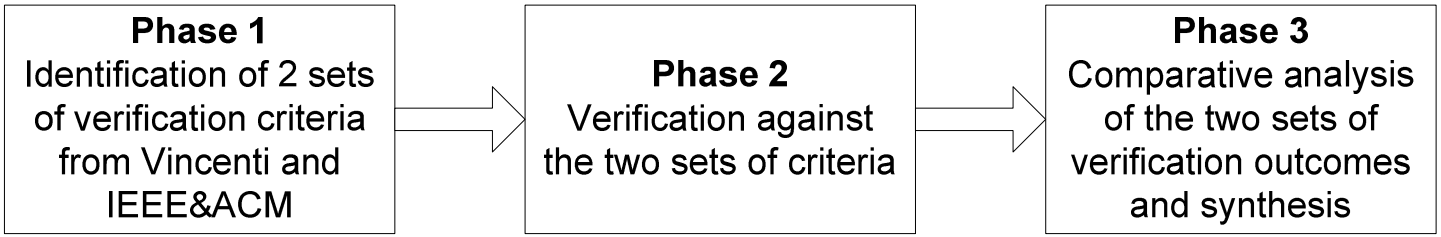
34 candidates (Séguin 2006)

	Candidates – In alphabetical order
1	Align incentives for developer and customer
2	Apply and use quantitative measurements in decision making
3	Build software so that it needs a short user manual
4	Build with and for reuse
5	Define software artifacts rigorously
6	Design for maintenance
7	Determine requirements now
8	Don’t overstrain your hardware
9	Don’t try to retrofit quality
10	Don’t write your own test plans
11	Establish a software process that provides flexibility
12	Fix requirements specification error now
13	Give product to customers early
14	Grow systems incrementally
15	Implement a disciplined approach and improve it continuously
16	Invest in the understanding of the problem
17	Involve the customer

Candidate principles - Ctd

No.	Candidates– In alphabetical order
18	Keep design under intellectual control
19	Maintain clear accountability for results
20	Produce software in a stepwise fashion
21	Quality is the top priority; long term productivity is a natural consequence of high quality
22	Rotate (high performer) people through product assurance
23	Since change is inherent to software, plan for it and manage it
24	Since tradeoffs are inherent to software engineering, make them explicit and document it
25	Strive to have a peer, rather than a customer, find a defect
26	Tailor cost estimation methods
27	To improve design, study previous solutions to similar problems
28	Use better and fewer people
29	Use documentation standards
30	Write programs for people first
31	Know software engineering's techniques before using development tools
32	Select tests based on the likelihood that they will find faults
33	Choose a programming language to assure maintainability
34	In face of unstructured code, rethink the module and redesign it from scratch.

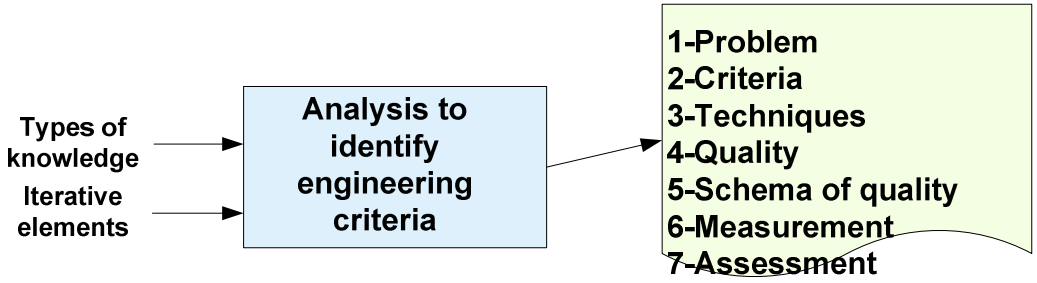
Methodology



Methodology

Phase 1: Identification of Engineering Criteria

Vincenti criteria

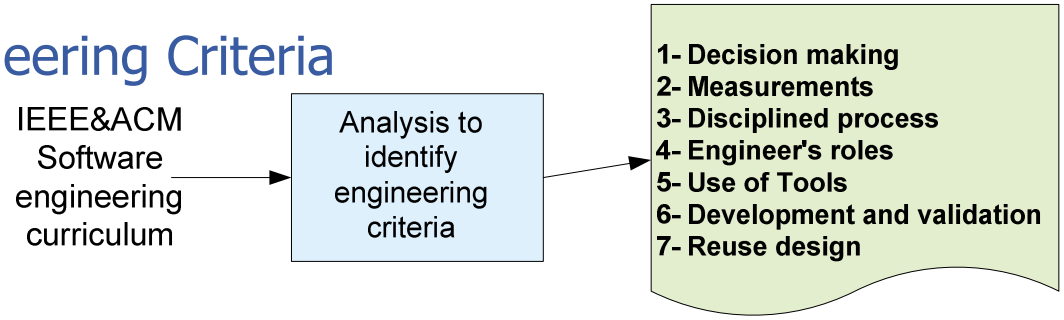


ID.	Vincenti's Engineering Criteria	Abbreviation
1	**Recognition of a problem**	**Problem**
2	**Identification of concepts and criteria**	**Criteria**
3	**Development of instrument and techniques**	**Techniques**
4	**Growth and refinement of opinion regarding desirable qualities**	**Quality**
5	**Combination of partial results from 2, 3 and 4 into practical schema for research**	**Schema of quality**
6	**Measurement of characteristics**	**Measurement**
7	**Assessment of results and data**	**Assessment**

Methodology Cont.

Phase 1: Identification of Engineering Criteria

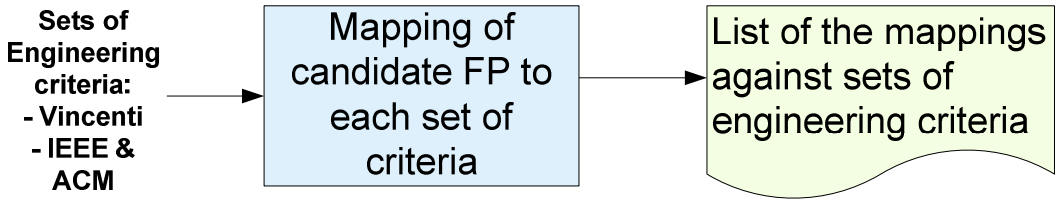
IEEE & ACM Engineering criteria



ID	Engineering Criteria Identified	Abbreviation
1	**Engineers proceed by making a series of decisions, carefully evaluating options, and choosing an approach at each decision-point that is appropriate for the current task in the current context. Appropriateness can be judged by tradeoff analysis, which balances costs against benefits.**	**Decision making**
2	**Engineers measure things, and when appropriate, work quantitatively; they calibrate and validate their measurements; and they use approximations based on experience and empirical data.**	**Measurements**
3	**Engineers emphasize the use of a disciplined process when creating a design and can operate effectively as part of a team in doing so.**	**Disciplined process**
4	**Engineers can have multiple roles: research, development, design, production, testing, construction, operations, management, and others such as sales, consulting, and teaching.**	**Engineer's roles**
5	**Engineers use tools to apply processes systematically. Therefore, the choice and use of appropriate tools is key to engineering.**	**Use of Tools**
6	**Engineers, via their professional societies, advance by the development and validation of principles, standards, and best practices.**	**Development and validation**
7	**Engineers reuse designs and design artifacts.**	**Reuse design**

Methodology Cont.

Phase 2: Verification against the two sets of criteria



Result of the mapping of the candidate FP to Vincenti engineering criteria

		C#1. Problem	C#2. Criteria	C#3. Techniques	C#4. Quality	C5. Schema of quality	C6. Measurement	C7. Assessment
1	**Align incentives for developer and customer**	**I**	**I**					
2	**Apply and use quantitative measurements in decision making**		**I**	**I**	**I**	**I**	**D**	**I**
3	**Build software so that it needs a short user manual**	**I**	**I**	**I**	**I**	**I**	**I**	**I**
4	**Build with and for reuse**		**D**	**I**	**I**	**I**		**I**
5	**Define software artifacts rigorously**	**I**	**I**	**I**	**I**	**I**	**I**	**I**
6	**Design for maintenance**	**I**	**I**	**I**	**I**	**I**		
7	**Determine requirements now**	**D**	**I**	**I**	**I**	**I**		
8	**Don't overstrain your hardware**	**I**	**I**				**I**	**I**
9	**Don't try to retrofit quality**	**I**	**I**	**I**	**D**	**D**	**I**	**I**

Mapping of the candidate FP to IEEE & ACM engineering criteria

		C1. Decision making	C2.M Measurements	C3. Disciplined process	C4 Engineer's roles	C 5 Use of Tools	C6.Dev elopment & validation	C 7. Reuse design
1	**Align incentives for developer and customer**			**I**				
2	**Apply and use quantitative measurements in decision making**	**D**	**D**					
3	**Build software so that it needs a short user manual**	**I**	**I**	**I**				
4	**Build with and for reuse**			**I**				**D**
5	**Define software artifacts rigorously**		**I**	**D**				
6	**Design for maintenance**	**I**		**I**	**D**			
7	**Determine requirements now**	**I**		**I**				
8	**Don't overstrain your hardware**	**I**	**I**					
9	**Don't try to retrofit quality**	**I**	**I**	**I**		**I**		

Methodology Cont.

Phase 3: Analysis and Consolidation using both sets of criteria

The analysis across each set of engineering criteria can then be grouped into 3 sets of candidate FP:

- with Vincenti mapping similar to the IEEE-ACM mapping;
- with Vincenti mapping with no equivalent IEEE-ACM mapping;
- with IEEE-ACM mapping with no equivalent Vincenti's mapping.

Candidate FP that meets directly criteria from either set

#	Vincenti Mapping	#	IEEE and ACM Mapping
2	Apply and use quantitative measurements in decision making	2	Apply and use quantitative measurements in decision making
4	Build with and for reuse	4	Build with and for reuse
		5	Define software artifact rigorously
		6	Design for maintenance
7	Determine requirements now		
9	Don't try to retrofit quality		
		12	Fix requirements specification error now
14	Grow systems incrementally		
15	Implement a disciplined approach and improve it continuously	15	Implement a disciplined approach and improve it continuously
16	Invest in the understanding of the problem		
		18	Keep design under intellectual control
21	Quality is the top priority; long term productivity is a natural consequence of high quality	21	Quality is the top priority; long term productivity is a natural consequence of high quality

List of fundamental principles of software engineering

#	Vincenti, IEEE and ACM mapping	
1	**Apply and use quantitative measurements in decision making**	**2**
2	**Build with and for reuse**	**4**
3	**Grow systems incrementally**	**14**
4	**Implement a disciplined approach and improve it continuously**	**15**
5	**Invest in the understanding of the problem**	**16**
6	**Quality is the top priority; long term productivity is a natural consequence of high quality**	**21**
7	**Since change is inherent to software, plan for it and manage it**	**23**
8	**Since tradeoffs are inherent to software engineering, make them explicit and document it**	**24**
9	**To improve design, study previous solutions to similar problems**	**27**

Hierarchy of Principles

#	Direct mapping to Vincenti criteria	Derived instantiation (= Indirect mapping)
2	**Apply and use quantitative measurements in decision making**	**18** Keep design under intellectual control **26** Tailor cost estimation methods
4	**Build with and for reuse**	**6** Design for maintenance **11** Establish a software process that provides flexibility **18** Keep design under intellectual control
14	**Grow systems incrementally**	**11** Establish a software process that provides flexibility
15	**Implement a disciplined approach and improve it continuously**	**1** Align incentives for developer and customer **17** Involve the customer **18** Keep design under intellectual control **5** Define software artifacts rigorously **20** Produce software in a stepwise fashion **31** Know software engineering's techniques before using development tools **19** Maintain clear accountability for results **29** Use documentation standards
16	**Invest in the understanding of the problem**	**7** Determine requirements now **12** Fix requirements specification error now
21	**Quality is the top priority; long term productivity is a natural consequence of high quality**	**9** Don't try to retrofit quality **22** Rotate (high performer) people through product assurance **25** Strive to have a peer, rather than a customer, find a defect
23	**Since change is inherent to software, plan for it and manage it**	**6** Design for maintenance **33** Choose a programming language to assure maintainability **32** Select tests based on the likelihood that they will find faults **34** In face of unstructured code, rethink the module and redesign it from scratch. **11** Establish a software process that provides flexibility
24	**Since tradeoffs are inherent to software engineering, make them explicit and document it**	**5** Define software artifacts rigorously **29** Use documentation standards
27	**To improve design, study previous solutions to similar problems**	**18** Keep design under intellectual control

References

- ACM, I.a., Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering, A.V.o.t.C.C. Series, Editor. 2004.
- Pierre Bourque, Robert Dupuis, Alain Abran, James W. Moore, Leonard Tripp, and S. Wolff, Fundamental principles of software engineering – a journey. Journal of Systems and Software, 2002. 62: p. 59-70.
- Jabir, Moore, J.W., Abran, A., Bourque, P., Dupuis, R., Hybertson, D., Jacquet, J.-P., Köller, A., Lowry, E., and Tripp, L.L., A Search for Fundamental Principles of Software Engineering – Workshop Report - Forum on Software Engineering Standards Issues, Montréal, Quebec, Canada, 21-25 October 1996, Computer Standards and Interfaces, vol. 19, pp. 155-160, 1998.
- Séguin, N., Inventaire, Analyse et Consolidation des Principes Fondamentaux du Genie Logiciel. 2006, Université du Québec à Montréal: Montreal.
- Séguin, N. and A. Abran, Inventaire des principes du génie logiciel. Revue Génie Logiciel, 2007: p. 45-51.
- Séguin, N. and A. Abran, Software Engineering Principles: A Survey and Analysis.
- Alain Abran, N.S., Pierre Bourque, Robert Dupuis. The search for software engineering principles: An overview of results. in PRInciples of Software Engineering. 2004.
- Boehm, B.W., Seven Basics Principles of Software Engineering. Journal of Systems and Software, 1983. 3(no 1): p. 366-371.
- Davis, A.M., 201 Principles of Software Development. 1995, New-York: McGraw-Hill.

References Cont.

- Buschmann, F., R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, Pattern Oriented Software Architecture. England. Wiley ed. 1996, England. 476 pages.
- Ghezzi, C., M. Jazayeri, and D. Mandrioli, Fundamentals of Software Engineering. 2th ed ed. 2003, New-Jersey: Prentice Hall.
- Dupuis, R., P. Bourque, A. Abran, and M.J. W. Principes Fondamentaux du génie logiciel : Une étude Delph, Le génie logiciel et ses applications, CNAM. 1997, Paris, Dec. 3-5 2007.
- Vincenti, W.G., What engineers know and how they know it, ed. M.R. Smith. 1990, Baltimore, London: The johns Hopkins University Press.
- Alain Abran, Luigi Buglione, and A. Sellami. Software measurement body of knowledge initial validation using Vincenti's classification of engineering knowledge types. in 14th International Workshop on Software Measurement - IWSM-MetriKon 2004. 2004. Konigs Wusterhausen, Germany: Shaker-Verlag.
- A. Abran, P.B., R. Dupuis and JW Moore & L. Tripp., The Guide to the Software Engineering Body of Knowledge - SWEBOK 2004 version. 2005, Los Alamitos: IEEE Computer Society Press.
- Abran, A. and K. Meridji, Analysis of Software Engineering from An Engineering Perspective. European Journal for the Informatics Professional, February 2006. 7(1): p. 46-52.

Questions

Discussion

- Do you agree with the 2 sources of information used?
- Do you know of other sources of information to define engineering criteria?
- For you, which engineering criteria are the most important?
- Do you agree with the direct and indirect mapping to the two sets of engineering criteria used?
- If not, according to you which one does not have the correct mapping?

Discussion Cont.

- In the list of the 9 principles selected, is there one with which you do not agree?
- Is there a missing principle in this list of the selected 9 principles that should be considered as a fundamental principle?
- In the hierarchy for the other 25 candidates, which one you do not agree (e.g. positioning in the hierarchy)?