Software Engineering from an Engineering Perspective: SWEBOK as a Study Object

Alain Abran[a,b], Kenza Meridji[b], Javier Dolado[a]
a Universidad del País Vasco/Euskal Herriko Unibertsitatea
b Ecole de technologie supérieure - Université du Québec

Abstract

Software engineering, as a discipline, is not yet as mature as other engineering disciplines and it lacks criteria to assess, from an engineering perspective, the current content of its body of knowledge as embedded in the SWEBOK Guide. What is then the engineering knowledge that should be embedded within software engineering? Vincenti, in his book 'What engineers know and how they know it' has proposed a taxonomy of engineering knowledge types. To investigate software engineering from an engineering perspective, these Vincenti's categories of engineering knowledge are used to identify relevant engineering criteria and their presence in SWEBOK.

Keywords – Software Engineering, SWEBOK, ISO 19759, Vincenti, Engineering Knowledge Types

1. Introducción

"Engineering is a problem-solving activity…dealing mainly with practical problems" (Vincenti [6]).

1.1. Overview

Software engineering (SE) 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) [3]. Of course, software engineering when compared to mechanical and electrical engineering is still an emerging engineering discipline not as mature as other classical engineering disciplines.

Much of the research work in software engineering has focused to date on developing methods, techniques and tools; much less research work has been carried out on:
exploring the engineering foundations of software engineering, including identifying the software engineering fundamental principles (FP), and next investigating on to apply them in research and practice.
Developing an international consensus on the software engineering body of knowledge and next ensuring a comprehensive coverage from an engineering perspective.

1.2. SE body of knowledge

Achieving consensus by the profession on a core body of knowledge is a key milestone in all disciplines, and has been identified by the IEEE Computer Society as crucial for the evolution of software engineering towards professional status. The Guide to the Software Engineering Body of Knowledge (SWEBOK Guide) [2,4]. The content of each knowledge area in the 2004 version of SWEBOK Guide was developed by domain experts and extensively reviewed by an international community of peers. This Delphi-type approach, while very extensive and paralleled by national reviews at the ISO level, did not specifically address the engineering perspective, nor did it provide a structured technique to ensure the completeness and full coverage of fundamental engineering topics. Therefore, it did not provide sufficient evidence that it had adequately tackled the identification and documentation of the knowledge expected to be present in an engineering discipline.

For the next update of the SWEBOK Guide research work has been initiated to analyze the content of the SWEBOK Guide in a structured

way in order to understand to which extent it does indeed include knowledge types typical of engineering disciplines, and to identify engineering knowledge could be missing. A challenge of course consists in figure out the criteria to be verified from an engineering perspective since, in the traditional engineering disciplines, such criteria have not been explicitly described in the generic engineering literature.

This paper presents an approach to identify engineering criteria to support such research needs. This paper is organized as follows: Section 2 introduces Vincenti's engineering viewpoint, and section 3 presents a set of models developed to facilitate the use of Vincenti's concepts for the analysis of an engineering discipline. Section 4 comments on a mapping of Vincenti's engineering design concept to the SWEBOK Guide software engineering design concept and on the Quality knowledge area. Section 5 presents a summary and future research directions.

2. Vincenti's Engineering Viewpoint

2.1. Overview and context

Vincenti, in his book [6], *What engineers know and how they know it*, proposed a taxonomy of engineering knowledge based on the historical analysis of five case studies in aeronautical engineering covering a roughly fifty-year period. He identified different types of engineering knowledge and classified them in six categories:

1 - Fundamental design concepts,
2 - Criteria and specifications,
3 - Theoretical tools,
4 - Quantitative data,
5 - Practical considerations, and
6 - Design instrumentalities.

Furthermore, Vincenti stated that this classification is not specific to the aeronautical engineering domain, but can be transferred to other engineering domains. However, he did not provide documented evidence of this applicability and generalization to other engineering disciplines, and no author could be identified as having attempted to do so either.

Vincenti provides a categorization of engineering design knowledge and the activities that generate it. However, the divisions are not entirely exclusive; some items of knowledge can contain the knowledge of more than one category. From Vincenti's definitions of each engineering knowledge-type category, a number of characteristics were identified; the goals of each category have also been identified, and these are listed in Table 1.

2.2. Related work

Maiebaum [5] was one of the first to identify the potential usefulness of Vincenti work for software engineering. However, since this classification of engineering knowledge had not been used to analyze other engineering disciplines, Abran & Meridji [1] modeled the embedded knowledge types descriptions for a partial analysis of the SWEBOK Guide. In particular, they investigated the engineering design concepts since at first glance there seemed to be a disconnect between the SWEBOK Guide design concept and Vincenti's description of engineering design. This is discussed in the next section.

3. Modeling of engineering knowledge

3.1. Overview

In [1] it was observed that Vincenti's categories are not mutually exclusive: it is therefore important to understand the relationships between them. Their initial modeling of Vincenti's categories of engineering knowledge is presented in Figure 1. This figure illustrates that, in seeking a design solution, designers move up and down within categories, as well as back and forth from one category to another.

Table 1. Vincenti: Engineering knowledge categories and goals

Engineering Knowledge Category	Goals
Fundamental design concepts	Designers embarking on any ***normal design*** bring with them ***fundamental concepts*** about the device in question.
Criteria and specification	To design a device embodying a given operational principle and normal configuration, the designer must have, at some point, ***specific requirements*** in terms of hardware.
Theoretical tools	To carry out their design function, engineers use a wide range of ***theoretical tools***. These include intellectual concepts as well as mathematical methods.
Quantitative data	Even with fundamental concepts and technical specifications at hand, mathematical tools are of little use without ***data*** for the ***physical properties*** or other ***quantities*** required in the formulas. Other kinds of data may also be needed to ***lay out details of the device*** or to ***specify manufacturing processes*** for production.
Practical considerations	To complement the theoretical tools and quantitative data, which are not sufficient. Designers also need ***less sharply defined considerations*** derived from experience.
Design instrumentalities	Besides the analytical tools, quantitative data and practical considerations required for their tasks, designers need to know ***how to carry out those tasks***. How to ***employ procedures*** productively constitutes an essential part of design knowledge.

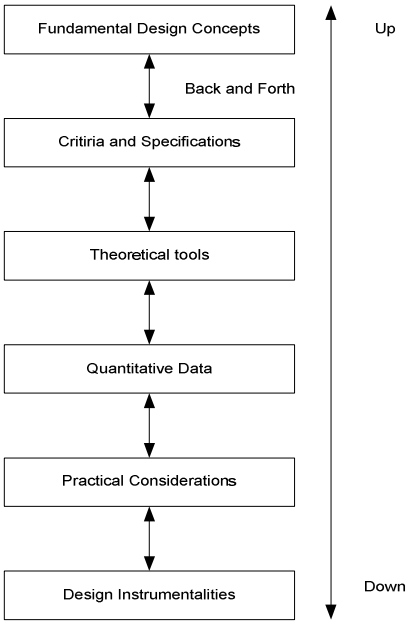

Figure 1. Vincenti's classification of engineering knowledge

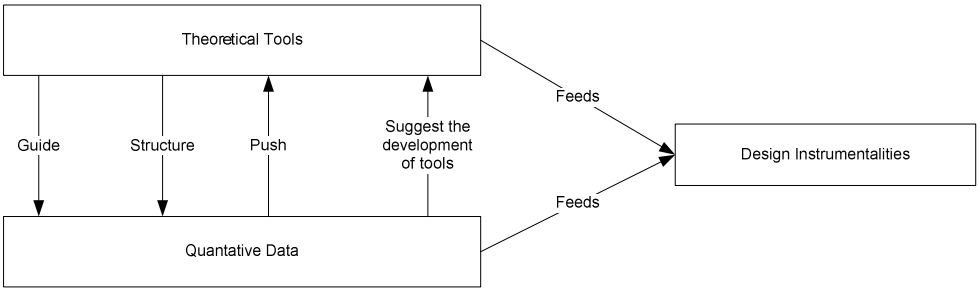

Figure 2: Relationships between theoretical tools & quantitative data

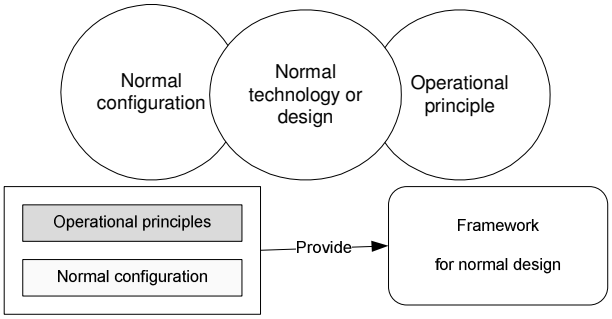

Figure 3: Relationships between normal configuration, operational principles & normal design

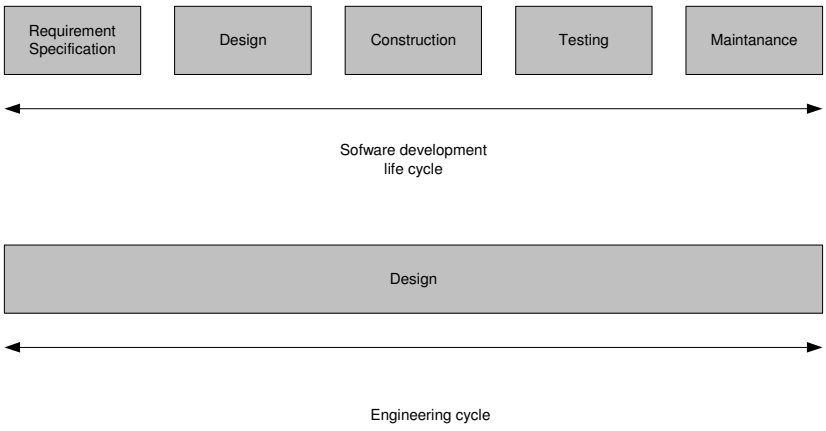

Figure 4. Design according to Vincenti vs. design in the software engineering life cycle

It was also noted that three categories (theoretical tools, quantitative data and design instrumentalities) are related in the following manner: theoretical tools guide and structure the data, while quantitative data suggest and push the development of tools for their presentation and application – see Figure 2. Furthermore, both theoretical tools and quantitative data serve as input for design instrumentalities, while appropriate theoretical tools and quantitative data are needed for technical specifications. This section presents some of their models to illustrate the relationships across these engineering concepts.

3.2. Fundamental design concepts

The goal of 'fundamental design concepts', according to Vincenti, is as follows: "*Designers setting out on any normal design bring with them fundamental concepts about the device in question,"* which means the definition of fundamental concepts related to the device by the designer. Fundamental design elements are composed of four elements; operational principles, normal configuration, normal technology and concepts in people's minds. At first, these concepts exist only in the designer's mind:

– **Operational principles** define the essential fundamental concept of a device. "How its characteristic parts… fulfill their special functions in combination to [sic] an overall operation which archives the purpose." The operational principles must be known by the designers first and constitute the basic components for the design, whereas operational principles are abstract, and the design moves from abstract concepts to precise concepts.

– **Normal configuration** is "the general shape and arrangement that are commonly agreed to best embody the operational principle."

– Normal technology is "*the improvement of the accepted tradition or its application under new or more stringent conditions.*" Design, in Vincenti, "denotes both the content of a set of plans (as in the design for a new airplane) and the process by which those plans are produced." There are two types of design: **normal design** and radical design. The latter is a kind of design that is unknown to the designer, and where the designer is not familiar with the device itself. The designer does not know how the device should be arranged, or even how it works. The former is a traditional design, where the designer knows how the device works. The designer also knows the traditional features of the device. This type of design is also the design involved in normal technology, which was mentioned earlier. In conclusion, "*normal design is evolutionary rather than revolutionary.*" Finally, a normal configuration and operational principles together provide a framework for normal design – Figure 3. In Vincenti, a normal technology, or design, is part of a normal configuration and of a related operational principle.

3.3. Criteria and specifications

The goal for 'criteria and specifications' can be expressed as follows: "*To design a device embodying a given operational principle and normal configuration, the designer must have, at some point, specific requirements in terms of hardware.*" The designer designs a device meeting specific requirements which include a given operational principle as well as a normal configuration. At first, the design problem must be well defined. Then, the designer translates general quantitative goals into specific quantitative goals: the designer assigns values or limits to the characteristics of the device which are crucial for engineering design. This allows the designer to provide the details and dimensions of the device that will be given to the builder. Furthermore, the output at the problem definition level is used, in turn, as input to the remaining design activities that follow. These specifications are more important where safety is involved, as in the case of aeronautical devices. The criteria on which the specifications are based become part of the accumulating body of knowledge about how things are done in engineering.

3.4. Theoretical tools

Theoretical tools are used by engineers to carry out their design. The goal of the 'theoretical tools' category is expressed by Vincenti as follows: "*To carry out their design function, engineers use a wide range of* ***theoretical tools****. These include intellectual concepts as well as mathematical*

methods". Intellectual concepts (such as design concepts, mathematical methods and theories) are tools for making design calculations. Both design concepts and methods are part of science.

In the first class of theoretical tools are mathematical methods and theories composed of formulas, either simple or complex, which are useful for quantitative analysis and design. This scientific knowledge must be reformulated to make it applicable to engineering. The engineering activity requires that thoughts be conceived in people's minds.

In the second class of theoretical tools are intellectual concepts, which represent the language expressing those thoughts in people's minds. They are employed first in the quantitative conceptualization and reasoning that engineers have to perform before they carry out the quantitative analysis and design calculations, and then again while they are carrying them out.

3.5. Quantitative data

The goal of 'quantitative data' is to lay down "the **physical properties** or other **quantities** required in the formulas. Other kinds of data may also be needed to **lay out details of the device** or to **specify manufacturing processes** for production." Besides fundamental concepts and technical specifications, the designers also need quantitative data to lay out details of the device. These data can be obtained empirically, or in some cases they can be obtained theoretically. They can be represented in tables or graphs.

These data are divided into two types of knowledge: prescriptive and descriptive.

- Descriptive knowledge is "*knowledge of how things are*." It includes physical constants, properties of substances and physical processes. In some situations, it refers to operational conditions in the physical world. Descriptive data can also include measurement of performance.
- Prescriptive knowledge is "*knowledge of how things should be to attain a desired end.*" An example might be: "In order to accomplish this or organize this, arrange things this way."

Operational principles, normal configuration and technical specifications are prescriptive knowledge, because they prescribe how a device should satisfy its objective.

3.6. Practical considerations

According to Vincenti, the goal of 'practical considerations' is "to complement the role of theoretical tools and quantitative data which are not sufficient. Designers also need for their work **less sharply defined considerations** derived from experience." This kind of knowledge is prescriptive in the way that it shows the designers how to proceed with the design to achieve it. Vincenti refers to practical considerations as constituting non codifiable knowledge derived from experience, unlike theoretical tools and quantitative data which are very precise and codifiable because these are derived from intentional research. This category of engineering knowledge is needed by designers as a complement to theoretical tools and quantitative data. These practical considerations are learned on the job, rather than at school or from books. They are not to be formalized or programmed. They are derived from design, as well as from production and operation. The practical consideration derived from production is not easy to define and cannot be codified, and a prototype is highly recommended to check the designer's work. An example of a practical consideration from operation is the judgment that comes from the feedback resulting from use.

3.7. Design instrumentalities

The goal of 'design instrumentalities' in the engineering design process required for the engineer's tasks is "*to know **how to carry out those tasks**. How to **employ procedure** productively constitutes an essential part of design knowledge.*" Having the analytical tools, quantitative data and practical considerations at hand, designers also need procedural knowledge to carry out their tasks, as well as to know how to employ these procedures.

Design instrumentalities contain instrumentalities of the process, the procedures, judgment and ways of thinking. The latter are less tangible than procedures and more tangible than judgment; an example of ways of thinking is 'thinking by

analogy'. Judgment is needed to seek out design solutions and make design decisions.

4. Analysis of the SWEBOK using engineering knowledge types concepts

4.1. The engineering design process in Vincenti

According to Vincenti, the engineering "design" concept "denotes both the content of a set of plans (as in the design for a new airplane) and the process by which those plans are produced." In Vincenti's view, design is an iterative and complex process which consists of plans for the production of a single entity, such as an airplane (device), how these plans are produced, and, finally, the release of these plans for production.

Vincenti mentions that there are two types of design in engineering, **normal** and **radical**. In the former, the designer knows how the device works, how it should be arranged and what its features are. In the latter, the device is new to the engineer who is encountering it for the first time. Therefore, the engineer does not know how it works or how it should be organized.

He also mentions that design is a multilevel and hierarchical process. The designer starts by taking the problem as input. The design hierarchies start from the project definition level, located at the upper level of the hierarchy where problems are abstracted and unstructured. At the overall design level, the layout and the proportions of the device are set to meet the project definition. At level 3, the project is divided into its major components. At level 4, each component is subdivided. At level 5, the subcomponents from level 4 are further divided into specific problems. At the lower levels, problems are well defined and structured. The design process is iterative, both up and down and horizontally throughout the hierarchy.

4.2. The engineering process and the Design concept in the SWEBOK Guide

The SWEBOK Guide is composed of ten knowledge areas, each represented by one chapter in the SWEBOK Guide. The Software Requirements KA (KA) is composed of four phases of software requirements: elicitation, analysis, specification and validation. The elicitation phase is the process of deriving requirements through observation of existing systems. Requirements specification is the activity of transforming the requirements gathered during the analysis activity into a precise set of requirements. Software Requirements Specifications describe the software system to be delivered. In the requirements validation phase, the requirements are checked for realism, consistency and completeness.

Software design is defined in [2,4] as both "the process of defining the architecture, components, interfaces, and other characteristics of a system or component" and "the result of [that] process." Software ***design*** in the software engineering life cycle is an activity in which software requirements are taken as input to the software design phase for analysis. "*Software requirements express the needs and constraints placed on a software product that contribute to the solution of some real-world problem."*

The result will be the description of the software architecture, its decomposition into different components and the description of the interfaces between those components. Also described will be the internal structure of each component and the related program.

4.3. Design KA: mapping between Vincenti and the SWEBOK Guide

The analysis of the term 'design' in both Vincenti and the SWEBOK Guide is presented in Table 4: it can be observed that it is defined significantly differently in the two documents, that is, design in engineering according to Vincenti is not limited to design as described in the SWEBOK Guide: in Vincenti, it goes far beyond the scope of the SWEBOK, that is: it is composed of the whole of the software engineering life cycle, as illustrated in Figure 8, whereas all the activities of software life cycle, like the requirements phase, the design phase, the construction phase and the testing phase) map to a single phase in the engineering cycle, that is, design. These activities do not necessarily take place in the same order: for instance, testing in engineering starts right at the beginning, at the problem definition level, and goes on until the final release of the plans for production, while in the software engineering life

cycle, as defined generically in the SWEBOK Guide, testing starts after the construction phase; on the other hand, the set of V&V concepts are spread out throughout the lifecycle in SWEBOK.

The detailed mapping between the different design levels in engineering and in the software engineering life cycle is presented in Table 2.

4.4. Identification of engineering concepts in the SWEBOK Software Quality KA

An analysis of the engineering content within the SWEBOK Guide using one of its ten KAs as a case study, that is, Software Quality is presented [1]. This analysis is based on the models of engineering knowledge described earlier. These models give us a very descriptive analysis of the various key elements contained in each of the corresponding engineering knowledge areas. This allows to make an appropriate mapping between the different categories of the engineering knowledge area and software quality. It helps in identifying the engineering elements contained in this topic, as well as the missing ones. As a result, it looks into the software quality area from an engineering perspective. Table 3 describes the mapping between the corresponding characteristics for the classification of engineering knowledge and the related software quality topics. This analysis can provide useful insights into possible strengths and weaknesses of the software quality topic: it helps categorize the knowledge contained in the Software Quality KA of the SWEBOK Guide: for instance, it covers all categories of engineering knowledge from an engineering viewpoint, but this does not mean that it is complete and inclusive.

5. Summary

Software engineering, as a discipline, is certainly not yet as mature as other engineering disciplines and it lacks well recognized fundamental principles, as well as criteria to assess, from an engineering perspective, the proposals put forward as statement of fundamental principles as well as the current content of its body of knowledge as embedded in the SWEBOK Guide. In this paper we have looked into an approach to identified engineering criteria that should be embedded within software engineering. In particular, we have looked at Vincenti at the taxonomy of engineering knowledge types proposed by Vincenti.

In particular, various models of the characteristics of the Vincenti's knowledge type have been illustrated. Next these concepts have been used to gain some insights into the some of the engineering concepts currently present and documented in the quality knowledge area of the SWEBOK Guide, but however labeled differently. The work presented here has involved investigating this engineering perspective, first by analyzing the Vincenti classification of engineering knowledge, and second by comparing the design concept in Vincenti vs. the design concept in the SWEBOK Guide.

The result of this analysis was to show that the design issue in Vincenti is not limited to the design issue in the SWEBOK Guide: Design in engineering according to Vincenti is not limited to design as described in the SWEBOK Guide: it goes beyond that, in that it is composed of the whole of the software engineering life cycle.

Finally, the SWEBOK Software Quality KA was selected as a case study and analyzed using the Vincenti classification as a tool to analyze this KA from an engineering perspective. This analysis was carried out to identify some of the strengths and weaknesses of the breakdown of topics for the Software Quality KA. It has shown that all the categories of engineering knowledge described by Vincenti are present in this KA of the SWEBOK; that is, it addresses the full coverage of all related engineering-type knowledge. This does not mean, however, that it is all-inclusive and complete, but only that the coverage extends to all categories of engineering knowledge from an engineering viewpoint.

The next stage of this R&D project will focus on investigating the application of Vincenti's engineering knowledge to the analysis of proposed software engineering principles.

Acknowledgements

This research project has been funded partially by the European Community's Sixth Framework Programme – Marie Curie International Incoming Fellowship under contract MIF1-CT-2006-039212.

Table 2. Mapping of the design process in engineering vs. the software engineering life cycle

Levels	Description of the design process in Vincenti engineering perspective	Corresponding set of concepts in SWEBOK
1	Project Definition	Requirements
2	Overall design – component layout of the airplane to meet the project definition.	Specification
3	Major component design – division of project into wing design, fuselage design, landing gear design, electrical system design, etc.	Architecture of the system
4	Subdivision of areas of component design from level 3 according to the engineering discipline required (e.g. aerodynamic wing design, structural wing design, mechanical wing design)	Detailed design
5	Further division of the level 4 categories into highly specific problems	Construction

Table 3: Quality concepts in the SWEBOK Guide using Vincenti's classification

Engineering Knowledge Category	Corresponding Characteristics	SWEBOK – quality related concepts
Fundamental design concepts	• About the design • Designers must know the operational principle of the device • How the device works • Normal configuration • Normal design • Other features may be (opened?)	• Planning the software quality process • Quality characteristics of the software (QI), (QE), (QIU) • Software quality models • Quality assurance process • Verification process • Validation process • Review process • Audit process
Criteria and specification	• Specific requirement of an operational principle • General qualitative goals • Specific quantitative goals laid out in concrete technical terms • The design problem must be "well defined". • Unknown or partially understood criteria • Assignment of values to appropriate criteria • This task takes place at the project definition level.	• Quality objective to be specified • Characteristics of quality tools • Software characteristics • Criteria for assessing the characteristics
Theoretical Tools	• Mathematical methods and theories for making design calculation • Intellectual concepts for thinking about design • Precise and codifiable	• Verification process model • Formal methods • Testing • Theory measurement • Verification/proving properties • TQM (Total Quality Management)

Quantitative data	• Specify manufacturing process for production • Display the detail for the device • Data essential for design • Obtained empirically • Calculated theoretically • Represented in tables or graphs • Descriptive knowledge • Prescriptive knowledge • Precise and codifiable	• Quality measurement • Experimental data • Empirical study • E.g. the process of requirement inspection • Value and cost of quality
Practical Considera-tions	• Theoretical tools and quantitative data are not sufficient. Designers also need considerations derived from experience. • It is are difficult to find them documented. • They are also derived from production & operation. • This knowledge is difficult to define. • Its defies codification • The practical consideration derived from operation is judgment. • Rules of thumb.	• Application quality requirements • Defect characterization
Design Instrumenta-lities	• Knowing how • Procedural knowledge • Ways of thinking • Judgment skills	• Quality assurance procedures • Quality verification procedures • Quality validation procedures • SQM process tasks & techniques • Management techniques • Measurement techniques • Project planning and tracking • Quality assurance process • Verification process • Validation process • Review process • Audit process

References

[1] Abran, A., Meridji, K., 'Analysis of Software Engineering from An Engineering Perspective', European Journal for the Informatics Professional ,vol. 7, No. 1, February , 2006 , pp. 46-52 . www.upgrade-cepis.org Upgrade: ISSN 1684-5285 Novática: ISSN 0211-2124

[2] Abran, A., Moore, J., Bourque, P., Dupuis, R., Tripp, L. (2005), Guide to the Software Engineering Body of Knowledge – SWEBOK, IEEE Computer Society Press, Los Alamitos, URL: http://www.swebok.org

[3] IEEE 610.12-1990 (1990), IEEE Standard Glossary of Software Engineering Terminology, Institute of Electrical and Electronics Engineers ISBN: 155937067X. 84 pages.

[4] ISO/IEC TR 19759-2005 (2005), Guide to the Software Engineering Body of Knowledge (SWEBOK), International Organization for Standardization - ISO, Geneva, 2005

[5] Maieubaum, T., 'Mathematical foundations of software engineering: a roadmap', 22nd International Conference on The future of Software Engineering, June 4 - 11, 2000, Limerick Ireland, Pages 161-172.

[6] Vincenti, W. G. (1990). *What engineers know and how they know it*. Baltimore, London: The Johns Hopkins University Press.