2012 38th Euromicro Conference on Software Engineering and Advanced Applications

Software Measurement in Software Engineering Education: A Delphi Study to Develop a List of Teaching Topics and Related Levels of Learning

Mónica Villavicencio[1,2]
[1]École de technologie supérieure, Montréal, Canada
[2] CIDIS-FIEC, Escuela Superior Politécnica del Litoral (ESPOL), Guayaquil, Ecuador
mvillavi@espol.edu.ec

Alain Abran[1]
[1]École de technologie supérieure
Montréal, Canada
alain.abran@etsmtl.ca

***Abstract*—Software engineering education is aimed at preparing professionals to achieve software quality, manage risk, and deliver projects on time and on budget. This paper reports on the findings of a Delphi study as part of a series of research studies designed to identify the software measurement topics that should be taught in software engineering programs. It also outlines the levels of learning and skills that are expected to be reached by undergraduate students.**

Keywords—software measurement; software engineering; higher education

I. INTRODUCTION

The software industry is facing a number of recurring issues, including achieving software quality, managing risk, and delivering projects on time and on budget, among others. Software engineering education is aimed at preparing professionals to tackle these issues through teaching software measurement. To date, there has been little research on the place of software measurement in software engineering education [1]. To fill this gap, a series of studies has been planned to gain insights into improving the teaching and learning of software measurement in software engineering programs at the undergraduate level.

In the exploratory phase, the researchers conducted a literature review, a Web survey, and a Delphi study. The Delphi study is still under way. The purpose of the literature review was to gain insights into how software measurement topics are taught in university courses, both in theory and in practice. The results of this review are reported in [2]. The Web survey was also designed to gather information about the way software measurement is taught in universities, and, in addition, to obtain information related to software process improvement (SPI) initiatives and measurement programs in companies specializing in the information technologies or in software development. The target audience for these studies is university teachers and software measurement practitioners. The preliminary results of the study are reported in [3], and a portion of the final results has become the subject of a recently submitted conference paper.

To complement these two studies, a Delphi study has been designed to contribute to the development of a consensus on the identification of the software measurement topics that should be taught in undergraduate programs and the required level of learning on those topics according to Bloom's taxonomy [4] (see Figure 1). This taxonomy is made up of six levels of learning, characterized by the following verbs: remember, understand, apply, analyze, evaluate, and create [4].

The complete Delphi study was designed to include a pilot test for improving the research instrument. This paper has two major sections: the overall methodology of the Delphi study; and the presentation of its preliminary findings.

The rest of this paper is organized as follows. Section II contains a brief explanation of the Delphi method, followed in section III by a description of the methodology used for our study. The results of the pilot test of the Delphi study are presented in section IV, and the conclusion in section V.

II. THE DELPHI METHOD

The Delphi method is an expert survey that is usually performed in several rounds. The method is characterized by a structured group communication process. Its objective is to obtain opinions from experts and agreement among them about a topic that is being investigated [5-10]. The Delphi method has several uses, some of which are: a) group problem solving; b) decision aiding; and c) forecasting [5-10]. This method has been used extensively in health care and educational research projects [5, 7, 9, 11, 12].

To ensure the effectiveness of the method, a number of considerations have to be taken into account. Some considerations were proposed in [9], of which the following seven were adopted in the design of our Delphi study: A) methodological choices, B) initial question, C) number of rounds, D) number of participants, E) expertise criteria, F) mode of interaction, and G) further verification. The other considerations refer to the chosen methods for data analysis and results reporting and publications, which were not completely developed by the time this pilot study was conducted. In the following section, an explanation of each consideration is given, along with the procedures used for conducting this Delphi study.

III. RESEARCH METHODOLOGY

A Delphi study requires the implementation of a series of steps, involving a literature review on the topic that is being investigated, followed by definition of the objective and of the research question. These definitions are presented next.

Objective: To identify the specific aspects of the software measurement topics that should be taught in undergraduate programs to achieve the required level of learning.

Research question: What specific software measurement topics should be emphasized in software engineering

978-0-7695-4790-9/12 $26.00 © 2012 IEEE
DOI 10.1109/SEAA.2012.14

education at the undergraduate level, and what level of learning is required according to Bloom's taxonomy?

need to justify the experts' responses by providing an explanation of their rationale.

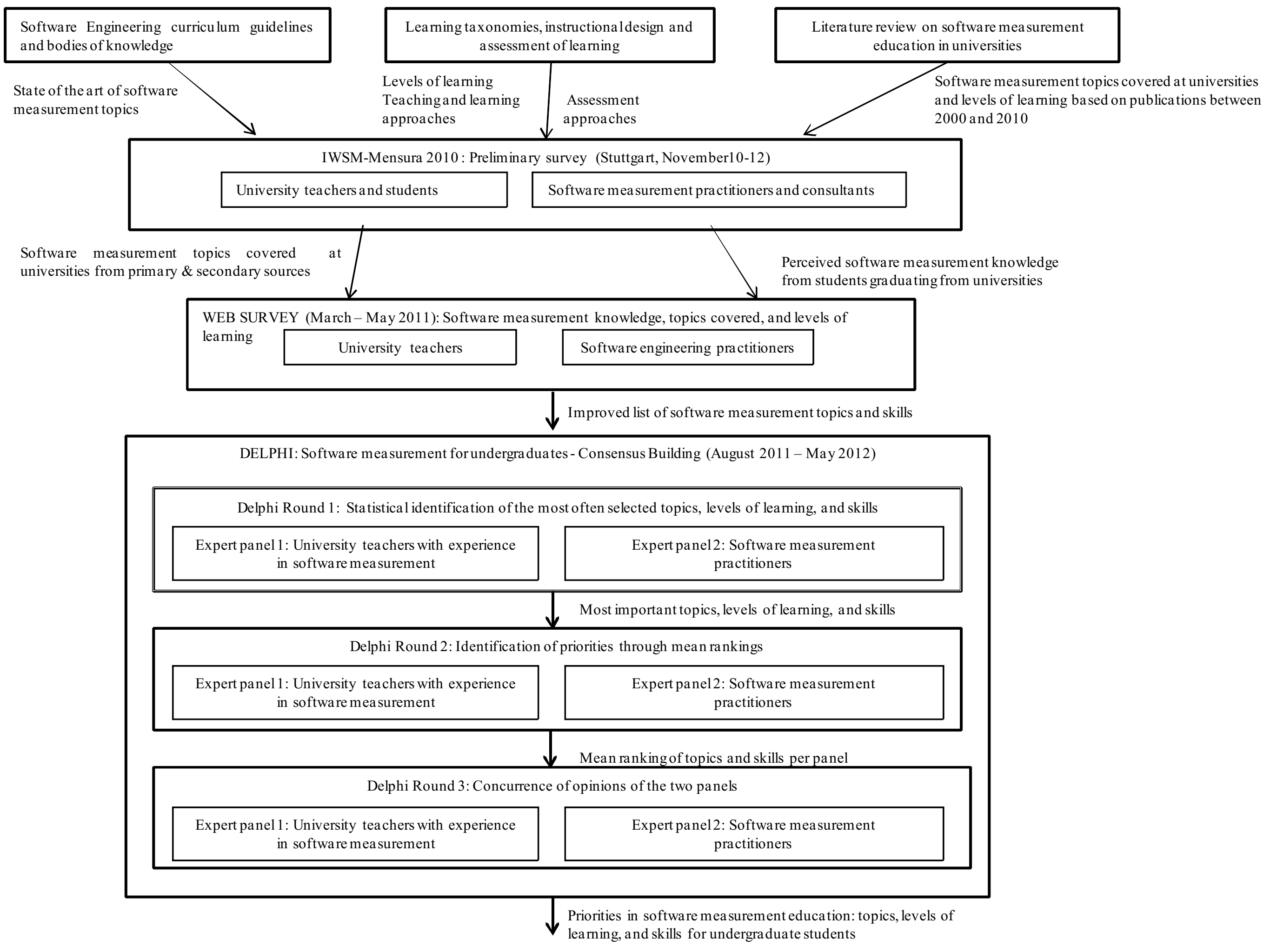
Software Engineering curriculum guidelines and bodies of knowledge
Learning taxonomies, instructional design and assessment of learning
Literature review on software measurement education in universities
State of the art of software measurement topics
Levels of learning
Teaching and learning approaches
Assessment approaches
Software measurement topics covered at universities and levels of learning based on publications between 2000 and 2010
IWSM-Mensura 2010 : Preliminary survey (Stuttgart, November 10-12)
University teachers and students
Software measurement practitioners and consultants
Software measurement topics covered at universities from primary & secondary sources
Perceived software measurement knowledge from students graduating from universities
WEB SURVEY (March – May 2011): Software measurement knowledge, topics covered, and levels of learning
University teachers
Software engineering practitioners
Improved list of software measurement topics and skills
DELPHI: Software measurement for undergraduates - Consensus Building (August 2011 – May 2012)
Delphi Round 1: Statistical identification of the most often selected topics, levels of learning, and skills
Expert panel 1: University teachers with experience in software measurement
Expert panel 2: Software measurement practitioners
Most important topics, levels of learning, and skills
Delphi Round 2: Identification of priorities through mean rankings
Expert panel 1: University teachers with experience in software measurement
Expert panel 2: Software measurement practitioners
Mean ranking of topics and skills per panel
Delphi Round 3: Concurrence of opinions of the two panels
Expert panel 1: University teachers with experience in software measurement
Expert panel 2: Software measurement practitioners
Priorities in software measurement education: topics, levels of learning, and skills for undergraduate students

Fig. 1. Methodology followed for building a consensus on the teaching of software measurement topics in undergraduate software engineering programs

We designed this study based on the seven considerations mentioned in the previous section, and followed the order of implementation of all the Delphi study phases proposed in [9].

A. Methodological choices: There are two possible choices for the Delphi study: a) quantitative analysis; and b) a combination of quantitative and qualitative analysis. For our study, option b) was more appropriate, because of the

B. Initial questions: There are two possibilities for stating initial questions: a) broad questions; and b) narrow questions. We selected the narrow questions option, as the two previous studies we conducted provided enough knowledge to develop a questionnaire with closed-ended questions. However, an explanation of the responses is requested in the new study.

C. Number of rounds: Based on the characteristics of this Delphi study, the authors believed that three rounds would be appropriate for identifying the relevant software measurement topics, since the study is characterized by the use of closed-ended questions and a homogeneous sample. In round 1, the list of software measurement topics that should be taught in undergraduate programs has to be verified by the Delphi participants. This list was developed based on the responses gathered in a previous study conducted by the researchers [3]. In addition, it took into account the measurement topics included in both the ACM and IEEE software engineering curriculum guidelines for undergraduate programs [13] and a software measurement body of knowledge [14]. Important to mention is that the list shows examples of the content of each topic in order to avoid misunderstandings among participants. In round 1, the expected level of learning per topic and the set of skills required for undergraduate students to complement their education in software measurement also have to be identified. To ensure the success of this round, a pilot test described in the next section was performed with the objective of testing the questionnaire to be used in the Delphi study.

In round 2, the five most important software measurement topics are expected to be identified by asking respondents to rank them in order of importance and to explain their selection. The importance of the topics will be identified by vote counting, with a median score of 50% or higher. This is an acceptable level of agreement, taking into consideration that software engineering is a relatively new engineering discipline and that it is expected to mature over time, like the more classical engineering disciplines. Round 3 is intended to verify whether or not the experts can agree on the list of consolidated priorities defined in round 2.

D. Number of participants: In a Delphi study, the characteristics of the sample play an important role. If it is homogeneous, 10 to 15 people are enough to reach agreement about a topic investigated. If the sample is heterogeneous, the number of participants should be much higher, perhaps several hundred people [9]. We consider our study sample to be homogeneous.

E. Expertise criteria: Software measurement expertise in teaching and/or professional experience is required for the participants in this study. The criteria differ according to the type of expert.

- Practitioners who qualify for the study will have the following profile: p1) Experience in software measurement through working on SPI programs, and/or on a software measurement team, and/or as a software measurement or SPI specialist; p2) Five or more years of professional experience in the field; p3) Published researcher in software measurement; p4) Post secondary education (Bachelor's degree, Master's or Ph.D.); and p5) Membership on a software measurement committee or in a software measurement association (not mandatory, but preferable).
- University teachers who qualify for the study will have the following profile: p1) Experience in the teaching of software measurement topics in software engineering courses, specialized software measurement courses, or related software engineering courses in which software measurement topics are covered; p2) Five or more years of teaching experience in topics related to software measurement; p3) Published researcher in software measurement; and p4) Membership on a software measurement committee or in a software measurement association (not mandatory, but preferable).

F. Mode of interaction: Paper-based questionnaires were used for the pre test and pilot test, and an online survey was devised for rounds 1, 2, and 3 [7, 9].

G. Further verification: In order to generalize results, they must be verified, which is usually accomplished by means of interviews or surveys [9]. For our study, the interview option has been chosen as it would enrich the results to be obtained in the Delphi study. Internationally recognized software measurement experts will participate after the third round in order to get their opinions about the priorities in software measurement education. We adopted the following criteria for a recognized expert: 1) Author of software measurement book(s) and/or article(s) in eminent journals; 2) More than 10 years of experience in the field; and 3) Experience in leading software measurement communities, associations, or enterprises.

Following the design of the study, an initial version of the questionnaire was developed and tested with eight Master's and Ph.D. students from the Department of Software Engineering and Information Technology at the École de technologie supérieure (ETS). Then, the design of the Delphi round 1 was refined and its pilot test was conducted based on the feedback provided by the students and two external reviewers. The findings of the pilot test are presented in the next section.

IV. THE PILOT TEST

To perform the pilot test, a workshop was organized at the ETS on August 4, 2011. A total of 25 people were invited and 19 of them attended, including 4 university teachers and 15 software practitioners. It took the participants 19 minutes, on average, to complete the questionnaire. During this period, they had the opportunity to provide suggestions for improving it.

Fifty percent of teachers had experience in teaching software engineering and 25% in teaching software measurement. With regard to the number of years of teaching experience, 75% had less than 5 years and only one had more than 20 years at the undergraduate level. The majority of practitioners had a graduate degree: 53% at the Master's level and 40% at the Ph.D. level. Eighty percent of the practitioners had less than three years of experience in software measurement, mostly gained working in software process improvement (47%) and/or as members of software measurement teams (27%). Only one practitioner had more than 25 years of experience in software measurement activities.

The remaining part of this section presents three tables that summarize the results of the pilot test. The first table presents the software measurement topics proposed for the undergraduate level. It can be observed in Table I that four topics reach a level of acceptance greater than 50%: B (The measurement process), D (Techniques and Tools), E (Measures for the requirements phase), and H (Basic concepts in software measurement). The explanations provided by the participants show that they all agree that undergraduates should be taught topics B and H, because these form the basis of software measurement. In addition, they consider that topic D is relevant because of its impact during the initial stage of any software project. The main reason provided for selecting topic E is that the accurate measurement of software/system size reduces the odds of exceeding either the scheduled time for project delivery or the budget allotted for the project, or both. These are considered the most common risks in the software industry. The majority of respondents emphasized that advanced topics related to this discipline should be taught in depth at the graduate level.

TABLE I. SOFTWARE MEASUREMENT TOPICS FOR UNDERGRADUATES

Software Measurement Topics	% of Preference		
	Teach. (n1)	*Pract. (n2)*	*All (n1+n2)*
A. Measures for the design phase	25	20	21
B. The measurement process	75	60	63
C. Measurement standards	50	40	42
D. Techniques and tools	25	67	58
E. Measures for the requirements phase	50	53	53
F. Measures for the construction phase	25	20	22
G. Measures for the testing phase	25	27	26
H. Basic concepts	100	87	90
I. Software management measures	50	27	32
J. Software quality measurement	50	27	32

n1=4; n2=15

Table II presents the desired levels of learning for the four topics cited as the most important in Table I (B, D, E, and H). It is important to note that, for each topic listed in Table I, a set of desired levels of learning was presented in the pilot test questionnaire. For example, topic D in Table II includes 5 desired levels of learning, from D1 to D5. D1 corresponds to the "remember" level of Bloom's taxonomy, D2 to D4 correspond to the "understand" level, and D5 corresponds to the "apply" level.

It is important to mention that the literature review revealed that the highest level of learning reported in the publications relating to the teaching of software measurement is "apply". As the authors were interested in finding out whether the experts consider the "analyze", "evaluate" and "create" levels of the Bloom's taxonomy are as important as the first three, they were also included in the questionnaire. However, the desired levels of learning per topic mainly selected by respondents fall into the first three levels of Bloom's taxonomy (remember, understand and apply); somehow confirming the findings of previous studies (see B2, D5 and H3).

Continuing with the explanation of Table II, agreement for the Bachelor's degree level was only reached in three of the topics (B2, D5, and H3). B2 and H3 correspond to level 2 of Bloom's taxonomy, which falls into the "Understand" category. The remaining topic (D5) corresponds to level 3, "Apply".

TABLE II. LEVELS OF LEARNING FOR UNDERGRADUATES IN PREFERRED SOFTWARE MEASUREMENT TOPICS

Desired Level of Learning	% of Preference		
	Teach (n1)	*Pract (n2)*	*All (n1+n2)*
The measurement process			
B1. Remembering the process	50	27	32
B2. Explaining the process	100	53	63
B3. Using the process in a particular project/situation	50	33	37
B4. Designing/modifying a process for a specific situation	50	13	21
Techniques and tools (T&T)			
D1. Remembering software measurement T&T	25	20	21
D2. Giving an example of T&T	25	20	21
D3. Comparing T&T	25	40	37
D4. Explaining T&T	50	33	37
D5. Using T&T in a particular project/situation	50	53	53
Measures for the requirements phase			
E1. Remembering the most common functional size measurement methods	25	20	21
E2. Interpreting the most common functional size measurement methods	100	20	16
E3. Obtaining the functional size of software in a particular project	75	40	47
E4. Analyzing software size measurement results	25	13	16
Basic concepts in software measurement			
H1. Remembering software measurement terminology, concepts, methods, and models	75	33	42
H2. Giving examples of basic concepts	75	33	42
H3. Explaining the basic concepts	75	53	58
H4. Using the basic concepts in a particular project/situation	50	40	42
H5. Designing/modifying measurement methods/procedures	25	13	16

n1=4; n2=15

Table III shows a list of skills that can be instilled in undergraduates to complement their education in software measurement. These skills are considered in the Delphi study because the development of skills in software engineering students is as important as the acquisition of new knowledge. The list of skills is based on several sources related to software engineering education that are summarized in [15].

The fourth column of Table III (% of preference of all participants) highlights the four skills considered as the most important to complement the teaching and learning of software measurement topics. These skills are: written communication, ethical practice, teamwork, and critical thinking. From this result, it is clear that practitioners participating in the pilot test had a remarkably high preference for the skill of critical thinking, which can be achieved when higher order levels of cognition are desired.

TABLE III. COMPLEMENTARY SKILLS TO BE INSTILLED IN UNDERGRADUATES

Skills to be instilled to complement software measurement education	**% of Preference**		
	Teach (n1)	***Pract (n2)***	***All (n1+n2)***
Oral communication	75	47	53
Written communication	75	67	68
Ethical practice	100	60	68
Team work	75	67	68
Leadership	0	0	0
Independent problem solving	0	47	37
Critical thinking	50	80	74

n1=4; n2=15

V. CONCLUSION

Up to now, based on the results of Table I, column 2, four topics have been identified as important for participants (more than 50% of preference): 1) The measurement process, 2) Techniques and Tools, 3) Measures for the requirement phase; and 4) Basic concepts in software measurement. It is important to highlight, however, that the opinions of teachers and practitioners differ to a certain extent in this pilot test. For example, practitioners put a special emphasis in topic D (Techniques and Tools) while teachers do not.

Another finding from the pilot test is that the levels of learning chosen were those located in the middle or at the end of each topic (see Table II), that is: B2, D5, and H3. For a better understanding, we have to refer to topics B (The measurement process), D (Techniques and tools), and H (Basic concepts in software measurement). We can see that the higher percentages correspond to "B2 Explaining the process" (63%, located in the middle), "D5 Using them in a given project/situation" (53%, located at the end) and "H3 Explaining them" (58%, located in the middle). These preferences imply that students must, for example, reach levels D1 to D4 before achieving level D5 ("Explain" and "Use"). It seems that the way this question was presented was not clear enough for the participants. This weakness is also evidenced by the fact that some of them only chose the higher level of learning, while others selected all the previous levels as well. To avoid this misunderstanding, the final version of the questionnaire will indicate that participants can choose all the possible levels of learning that may apply.

Similar to what was observed in Table I, considerable differences in opinions between teachers and practitioners became also apparent in the levels of learning as shown in Table II (see for example the levels B2, E2, E3, H1 and H2).

An interesting result from the pilot test is that a high percentage of the participants consider the skill of critical thinking as highly important in software measurement education. Instilling this skill in undergraduate students is a complex and challenging task for educators, and requires a paradigm shift in which the students become central to the learning process and the teachers become the facilitators of learning rather than evaluators of performance. In this paradigm, students would be exposed to active learning through the exercise of relevant in- and out-class activities involving more practical, real world problems, an approach in alignment with the constructivist view of learning [4, 16-18]. Owing to the perceived importance of critical thinking, an adjustment was made to the questionnaire developed for round 1 of the Delphi study. This adjustment considers the inclusion of higher levels of learning for each software measurement topic. We believe that this change will allow better matching of the levels of learning and the complementary skills.

Although the findings presented here may not be conclusive for purposes of enhancing the teaching and learning of software measurement, they provide initial evidence of what topics and what levels of learning should be emphasized in university courses and the skills that need to be instilled in students. Attention should be paid, however, that the sample of this pilot test was mainly composed of practitioners, which presents a validity threat for generalization purposes. Certainly, more research is required to find answers of how to enhance software measurement education at universities. In this direction, the methodology presented in this paper is valuable as it provides the complete guide for reaching the ultimate research objectives. The full Delphi study is planned for the first semester of 2012 to complement the findings of this pilot study, in which a similar amount of participants for each panel will be considered. At the time this paper was written, the round 2 of the Delphi study was in the design stage. As a final outcome on the series of studies mentioned on the present paper, an educational framework will be developed for improving the teaching and learning of software measurement at the undergraduate level.

REFERENCES

[1] C. Gresse von Wangenheim, et al., "Empirical evaluation of an educational game on software measurement," Empirical Software Engineering, vol. 14, pp. 418-452, 2009.

[2] M. Villavicencio and A. Abran, "Software Measurement in Software Engineering Education: A Comparative Analysis," in International

Conference on Software Measurement IWSM/MetriKon/Mensura, Stuttgart, Germany, 2010, pp. 633-644.

[3] M. Villavicencio and A. Abran, "Facts and Perceptions Regarding Software Measurement in Education and in Practice: Preliminary Results," Journal of Software Engineering and Applications vol. 4, pp. 227-234, 2011-04-29 2011.

[4] L. Anderson, et al., A taxonomy for learning, teaching and assessing. A revision of Bloom's taxonomy of Educational Objectives. New York: Addison Wesley Longman, Inc., 2001.

[5] T. Amos and N. Pearse, "The Delphi technique and educating entrepreneurs for the future," in 7th European Conference on Research Methodology for Business and Management Studies: Academic Publishing Limited, Reading, UK 2008 2008, pp. 17-24.

[6] P. Bourque, et al., "Fundamental principles of software engineering-a journey," Journal of Systems and Software, vol. 62, pp. 59-70, 2002.

[7] H.-L. Hung, et al., "Methodological and conceptual issues confronting a cross-country Delphi study of educational program evaluation," Evaluation and Program Planning, vol. 31, pp. 191-198, 2008.

[8] C. Okoli and S. D. Pawlowski, "The Delphi method as a research tool: an example, design considerations and applications," Information & Management, vol. 42, pp. 15-29, 2004.

[9] G. J. Skulmoski, et al., "The Delphi Method for Graduate Research," Journal of Information Technology Education, vol. 6, pp. 1-21, 2007.

[10] E. Hall, "The Delphi Primer: Doing Real-World or Academic Research Using a Mixed-Method Approach," in The Refractive Thinker®: Vol II Research Methodology vol. 2: The Lentz Leadership Institute, 2009, Kindle Locations 103-104.

[11] D. W. Gatchell, et al., "Determination of the core undergraduate BME curriculum -- the 1st step in a Delphi study," in Conference Proceedings. 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Sept. 1-5, 2004, Piscataway, NJ, USA, 2004, pp. 5200-5201.

[12] P. C. Howze and C. Dalrymple, "Consensus without all the meetings: using the Delphi method to determine course content for library instruction," Reference Services Review, vol. 32, pp. 174-84, 2004.

[13] IEEE ACM. (2004, Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. SE2004 Volume – 8/23/2004, 135. Available: http://sites.computer.org/ccse/SE2004 Volume.pdf

[14] A. Abran, et al., "Software Measurement Body of Knowledge," in Encyclopedia of Software Engineering vol. 1:1. London: Taylor & Francis, 2010, pp. 1157-1168.

[15] M. Villavicencio and A. Abran, "Educational Issues in the Teaching of Software Measurement in Software Engineering Undergraduate Programs," in Joint Conference of the 21st Int'l Workshop on Software Measurement and the 6th Int'l Conference on Software Process and Product Measurement (IWSM-MENSURA), Nara, Japan, 2011, pp. 239-244.

[16] J. Biggs and C. Tang, Teaching for quality learning at university, Third ed., Buckingham: Society for Research into Higher Education & Open University Press, 2007.

[17] A. Romiszowski, "Fostering skill development outcomes," in Instructional design theories and models. vol. III, C. Reigeluth and A. Carr-Chellman, Eds., New York: Routledge, Taylor & Francis Group, 2009, pp. 199-224.

[18] L. Lindsey and N. Berger, "Experiential approach to instruction," in Instructional-Design Theories and Models. vol. III, C. Reigeluth and A. Carr-Chellman, Eds., New York: Routledge, Taylor & Francis Group 2009, pp. 117-142.