

Facts and Perceptions Regarding Software Measurement in Education and in Practice: Preliminary Results

Mónica Villavicencio^{1,2}, Alain Abran¹

¹École de Technologie Supérieure, Montréal, Canada; ²CIDIS-FIEC, Escuela Superior Politécnica del Litoral (ESPOL), Guayaquil, Ecuador.

Email: mwillavi@espol.edu.ec, alain.abran@etsmtl.ca

Received March 15th, 2011; revised March 24th, 2011; accepted March 27th, 2011.

ABSTRACT

How is software measurement addressed in undergraduate and graduate programs in universities? Do organizations consider that the graduating students they hire have an adequate knowledge of software measurement? To answer these and related questions, a survey was administered to participants who attended the IWSM-MENSURA 2010 conference in Stuttgart, Germany. Forty-seven of the 69 conference participants (including software development practitioners, software measurement consultants, university professors, and graduate students) took part in the survey. The results indicate that software measurement topics are: 1) covered mostly at the graduate level and not at the undergraduate level, and 2) not mandatory. Graduate students and professors consider that, of the measurement topics covered in university curricula, specific topics, such as measures for the requirements phase, and measurement techniques and tools, receive more attention in the academic context. A common observation of the practitioners who participated in the survey was that students hired as new employees bring limited software measurement-related knowledge to their organizations. Discussion of the findings and directions for future research are presented.

Keywords: Software Measurement, Education, Software Engineering

1. Introduction

This paper is part of a series of research studies related to software engineering education and software measurement education in particular. The motivation behind these studies is to help organizations succeed in the implementation of software measurement programs as part of the adoption of software process improvement initiatives. In this respect, previous studies have shown that organizations frequently face difficulties in implementing this type of programs. One of the reasons seems to be that practitioners perceive software measurement as a complex task [1-4]. Another reason may be due to the lack of guidelines for conducting measurement programs [1,5,6] in organizations. Moreover, it has been identified that universities, in software engineering education, are not paying enough attention to measurement topics in the way they should [7]. However, these topics are explicitly included in the curriculum guidelines for undergraduate and graduate programs in software engineering [8,9].

To contribute in finding a solution to the problems

stated in the above discussion, one of the steps in these studies is to analyze university curricula and the way in which software measurement is taught in an academic environment. This requires that data be gathered from primary and secondary sources, first from a literature review and then through surveys, interviews, and a Delphi study. From these sources, it is expected that the following will be identified: 1) software measurement topics taught in university programs; 2) instructional design theories suitable for this specialized subject; and 3) useful guidelines for software measurement education.

A literature review was performed to gain insights into how software measurement topics are taught in university courses, both in theory and in practice. The review consisted of a content analysis of publications appearing between 2000 and 2010, in which experiments with students were reported by university teachers. From the set of reviewed publications, we identified how software measurement topics are taught, specifically when students are exposed to practical measurement activities during

the development of a toy or real project. Initial observations were reported in [10]. Since those findings were based only on secondary sources, the next step was to design a survey to obtain preliminary answers from primary sources to the following questions: How is software measurement addressed in undergraduate and graduate programs in universities? Do organizations consider that the graduating students they hire have an adequate knowledge of software measurement? The survey was promoted under the leadership of the IWSM- MENSURA 2010 [11] program committee, in which the attendees were asked to participate voluntarily. This survey is a preliminary study, designed to identify potential problems in the wording of questions and the structure of the questionnaires. In further studies, the authors will conduct a Web-based survey, in which the necessary adjustments will have been made.

This paper is organized as follows. Section 2 explains the survey methodology. Section 3 presents the survey findings, as well as a comparison of the respondents' answers. Section 4 describes the threats to the validity of this study, followed by the conclusions and future work in Section 5.

2. Survey Methodology

This section describes the process for designing and administering the IWSM-MENSURA 2010 survey, which involved three steps: 1) survey design; 2) pre-test; and 3) administration of the survey to the IWSM-MENSURA 2010 attendees who volunteered to take part in the survey.

We summarized the findings from the secondary sources reported in [10] in tables, and used them as inputs for the survey design. Four types of survey participants were identified, corresponding to the four types of questionnaires (Q) that were developed to collect relevant information from each party:

Q1: Teachers

Q2: Students

Q3: Practitioners

Q4: Consultants

Each type of questionnaire included a general information section to gather data related to the university department and/or program where software measurement topics were being taught. In this section, respondents were asked to indicate the type of software organization for which they were working or providing consulting services.

Q1 and Q2 were focused on collecting information on how software measurement is taught in university programs. The two questionnaires contained similar questions, so that the answers of professors and students could be compared and any common responses relating to an

academic context could be identified. For example, one of the questions asked teachers to select all the approaches generally used in class for teaching software measurement topics. In parallel, students were asked to "choose all the approaches that the teacher used in class when measurement topics were reviewed."

To identify the software measurement topics covered in university curricula, a set of questions was developed based on the proposal of a software measurement knowledge area being considered in the 2010-2011 review process of the SWEBOK Guide: teachers and students were asked to indicate the measurement topics addressed in curricula and the level of learning expected to be reached in each of them.

Q3 and Q4 were aimed at gathering information from practitioners relating to software process improvement initiatives being implemented in the organizations for which they were working. These covered: the type of software process improvement initiatives, the type of certification obtained, the standards used for the definition of collected measures, the importance of measurement in the decision making process, and the software measurement level of knowledge of new employees (university students).

Each of the four questionnaires was pre-tested with at least 3 individuals. From the answers and feedback provided by the respondents on the pre-test, we were able to: 1) review the number of answers to some of the questions; 2) provide examples in two of the questions of Q1 and Q2 to make them easier to understand; 3) reformulate some questions; and 4) split some questions into two or more questions, where needed.

Finally, the survey was publicly announced during the IWSM-MENSURA 2010 conference, and attendees were asked to participate voluntarily in the survey. Sixty-nine attendees received the questionnaires and 47 completed them. The number of respondents for each type of participant is as follows: Q1, 12 professors; Q2, 10 graduate students; Q3, 14 software practitioners; and Q4, 11 software measurement consultants. The sample size for each population is small; however, the data gathered in this initial survey are valuable and the respondents' answers will contribute to enhancing the survey questionnaires for further studies.

3. Survey Findings

This section summarizes the survey findings and discusses similarities and differences in participants' responses, including comparisons between the answers of teachers and students, and between those of practitioners and consultants. All the percentages included in the tables in this section have been rounded to the closest integer.

3.1. Findings from Q1 (Teachers) and Q2 (Students)

Table 1 shows the enrolment of respondents in university departments. The majority of teachers surveyed (58%) were from computer science departments and the majority of graduate students were from software engineering departments (40%). When teachers were asked about their domain of expertise, 84% mentioned that they were actively involved in teaching or doing research in software engineering, including software measurement, empirical software engineering and software construction. Eight percent of the remaining teachers indicated that was working on embedded systems, and the other 8% did not answer the question.

On the side of the students, fifty percent of those who were not from software engineering departments mentioned that they were enrolled in software engineering specializations. It must be noted that nine of the ten students were Ph.D. researchers in software engineering,

while the tenth was enrolled in a Master's degree program in software engineering.

Regarding the software measurement topics covered in university courses, **Table 2** presents a summary of the types of program in which the respondents are enrolled. Clearly, computer science programs with specializations in software engineering and software engineering programs are those that are more oriented to include measurement topics in their curriculum.

When teachers' and students' responses are compared regarding the software measurement topics covered in university programs, some differences can be observed. Teachers were mostly giving software engineering courses in undergraduate programs and software measurement courses in graduate programs, while students indicated that, in their experience, software measurement topics are mainly covered in graduate programs. This means that, based on the students' responses, software measurement is usually studied at the Master's level. As illustrated in **Table 3**, both students and professors agreed that software

Table 1. Enrolment of respondents in university departments.

Respondents	% Computer Science	% Software Engineering	% Information Systems	% Other
Teachers (n1)	58	17	8	17
Students (n2)	30	40	20	10

n1 = 12, n2 = 10

Table 2. University programs that include software measurement topics.

Courses in which software measurement topics are covered	% Computer science	% Software engineering	% Software engineering specialization (Comp Science)	% Information systems	% Other
Taught by professors (n1)	25	33	42	8	0
Received by students (n2)	10	50	20	10	10

n1 = 12, n2 = 10

Table 3. Courses that include software measurement topics.

Course focus	UNDERGRADUATE program			GRADUATE program			
	University Teachers' Responses (n1)						
	% Undergrad	Mandatory	Optional	% Graduate	Mandatory	Optional	
SW. engineering	42	80	20	25	100	0	
SW. measurement	8	0	100	42	50	50	
Other	25	33	67	8	0	100	
Course focus	Graduate Students' Responses (n2)						
	% Undergrad	%Mandatory	%Optional	% Graduate	Mandatory	Optional	
	SW. engineering	10	100	0	40	75	25
	SW. measurement	0	0	0	80	13	88
Other	0	0	0	10	0	100	

n1 = 12, n2 = 10

engineering courses are usually mandatory in undergraduate and graduate programs, while software measurement courses are optional. In this survey, a required course, *i.e.* taken by all students, was considered to be mandatory. In contrast, students can choose whether or not to take an optional course.

As mentioned in Section 2, a list of software measurement topics was developed based on the SWEBOK guide, chapter 12 [12]. The list is shown in **Table 4**, and the findings related to their level of learning in **Table 5**. The latter is presented as a matrix-type table, with the expected levels of learning in rows and the measurement topics covered in columns. The former are in accordance with Bloom's taxonomy, which lists the following six levels of learning [8,13]:

- Knowledge: Students are basically oriented towards remembering and recalling information.
- Comprehension: Students are able to understand and make use of the information they have received by describing, interpreting, and explaining it.
- Application: Students can properly apply concepts learned to a given or unexpected problem or situation.
- Analysis: Students can break down the subject of study into its parts and define the relationship between them.
- Synthesis: Students are able to create a new idea or product by using prior knowledge
- Evaluation: Students are able to make judgments about the value of materials, ideas, and so forth.

To facilitate analysis of the collected data, each topic is classified according to the teachers' and students' responses. As observed in **Table 5**, some coincidences exist among the answers provided by the two groups (see shaded areas). These findings seem to suggest that, based on Bloom's taxonomy, topics A and B are commonly learned at the knowledge and comprehension levels, and topics D, F, and L at the application level.

Other sections of questionnaires Q1 and Q2 were designed to reveal how software measurement is taught in class, the types of projects developed by students, and the measures usually collected in an academic environment. By examining the results derived from these sections, we find that 100% of teachers and students are of the opinion that software measurement is mainly taught through lectures, and more than 50% through case studies. In addition, in some courses, students are asked to develop toy or real projects in which size and total effort are collected as measures.

3.2. Findings from Q3 (Practitioners) and Q4 (Software Measurement Consultants)

As in the previous section, the answers from practitioners

Table 4. Software measurement topics included in Q1 and Q2.

ID	SOFTWARE MEASUREMENT TOPICS
A	BASIC CONCEPTS
B	MEASUREMENT PROCESS
C	MEASUREMENT STANDARDS
D	TECHNIQUES AND TOOLS
E	REPOSITORIES FOR QUANTITATIVE DATA
F	MEASURES FOR REQUIREMENTS PHASE
G	MEASURES FOR DESIGN PHASE
H	MEASURES FOR CONSTRUCTION PHASE
I	MEASURES FOR THE TESTING PHASE
J	MEASURES FOR THE MAINTENANCE PROCESS
K	CONFIGURATION MANAGEMENT MEASURES
L	SOFTWARE ENG. MANAGEMENT MEASURES
M	MEASURES RELATED TO QUALITY
N	OTHER

and consultants were arranged and presented in tables to compare their responses to the same question. **Table 6** shows the educational level of employees working in software organizations and software measurement consulting firms. It can be observed that the participants' organizations do not usually hire people with only a high-school diploma. Instead, they prefer to recruit people with Bachelor's and Master's degrees. To a lesser extent, people with Ph.D. degrees are hired by such organizations.

Practitioners and consultants were also surveyed, in order to determine how organizations assess the software measurement knowledge acquired by graduating students when they become their employees. They were asked to rank, from 'none' to 'more than expected', the measurement knowledge that they perceive students bring into their organizations. **Table 7** shows that respondents mostly believe that the holders of Bachelor's degrees have little or no knowledge of software measurement, and that employees with a Master's degree come to software organizations with little knowledge on this subject, or an amount that would normally be expected. It should be noted that the percentages presented in **Table 7** were calculated based on the number of answers collected, either for the Bachelor's or Master's degree level in each sub-group of respondents. This was done because some organizations did not have employees with Bachelor's or Master's degrees only, but with doctorates.

Practitioners and consultants were also asked, in a free

Table 5. Expected level of learning for software measurement topics.

Expected level of learning	SOFTWARE MEASUREMENT TOPICS																							
	A		B		C		D		E		F		G		H		I		L		M		N	
	Tch	Std	Tch	Std	Tch	Std	Tch	Std	Tch	Std	Tch	Std	Tch	Std	Tch	Std	Tch	Std	Tch	Std	Tch	Std	Tch	Std
Remember definitions	67	50	17	20	8	30	33	10	0	20	25	40	0	10	0	0	8	0	25	10	8	0	8	0
List elements	17	30	17	10	0	10	17	10	8	10	17	20	0	10	0	0	0	0	17	30	8	0	0	0
Recognize among a list	25	20	17	20	8	0	17	0	8	0	17	10	0	0	0	0	0	0	17	10	17	0	0	0
Give examples	17	30	8	20	0	10	25	0	0	10	25	10	0	0	0	0	0	0	17	0	8	0	0	0
Explain a concept	33	50	17	40	0	0	17	0	0	20	25	30	8	0	8	0	0	0	17	10	8	0	0	0
Tell differences & similarities	0	10	0	10	0	40	17	30	8	0	8	20	0	10	0	0	0	0	17	0	8	0	0	0
Apply what students learn in exercises	0	10	8	10	8	20	17	30	0	30	17	40	8	10	8	0	8	0	8	10	0	0	0	0
Use concepts, models. in a project	8	20	0	20	0	30	25	30	8	20	17	40	8	10	0	0	0	0	25	10	8	0	8	0
Differentiate components, relationships.	17	0	17	0	0	10	8	0	0	0	8	20	0	0	0	0	0	0	8	0	0	0	0	0

Tech = n1 = 12, Std = n2 = 10

Table 6. Educational level of employees.

% Employees	Software companies (n3)								Software measurement consultant firms (n4)							
	High School		Bachelor's		Master's		PhD		High School		Bachelor's		Master's		PhD	
	#	%	#	%	#	%	#	%	#	%	#	%	#	%	#	%
0%	7	50	3	21	1	7	2	14	8	73	7	64	2	18	5	46
<= 25%	3	21	3	21	3	21	11	79	2	18	1	9	1	9	2	18
> 25 < 50%	2	14	4	29	4	29	0	0	0	0	0	0	1	9	2	18
>= 50-75%	2	14	3	21	4	29	1	7	0	0	2	18	3	27	0	0
> 75%	0	0	1	7	2	14	0	0	1	9	1	9	4	36	2	18

n3 = 14 answers from practitioners working in software companies; n4 = 11 answers from measurement consultants

Table 7. Software measurement knowledge of new employees.

SW. Meas. Knowledge	Practitioners (n3)				Consultants (n4)			
	Bachelor's		Master's		Bachelor's		Master's	
More than expected	0.0%		7.7%		0.0%		0.0%	
Good	9.1%		7.7%		0.0%		11.1%	
Normal	18.2%		30.8%		0.0%		33.3%	
Little	54.5%		53.8%		71.4%		55.6%	
None	18.2%		0.0%		28.6%		0.0%	

n3 = 14, n4 = 11

format, for their opinions on why there is a lack of knowledge on software measurement. The answers to this question were as follows:

1) There is a gap between academic programs and the industrial environment.

2) Software measurement topics are included in software engineering courses, which are too general.

3) Software measurement is not a mandatory subject in university curricula.

4) Measurement is mostly taught in Master's degree programs.

5) Professors do not keep up to date, although some curricula are adequate in this respect.

6) Few measurement specialists are needed in the software industry.

An analysis of the reasons mentioned above reveals that these answers fairly closely match those provided by teachers and students, and reported in **Table 3**.

Another important survey observation is that most of the organizations in which the participants were working had software process improvement programs (SPI), CMMI being the most popular of these (practitioners 71%, consultants 55%), followed by in-house initiatives.

Moreover, to gain better insights into software measurement practices in organizations, two additional questions were included in the survey. The first is related to the standards used for measurement, and the second to the level of importance accorded to project management and maintenance measures. **Tables 8** and **9** present the results of these questions, and show that respondents used the ISO 20926 standard for functional size measurement, followed by ISO 19761. The percentages that appear in **Tables 8** and **9** were calculated based on the number of respondents who reported that there were software measurement programs in place in their organizations.

From **Table 9**, we can see that companies put significantly more emphasis on project management than on maintenance measures.

4. Threats to Validity

Two threats to the validity of our study have been identified, classified as internal or external.

In terms of internal validity, one threat has to do with the fact that the respondents may not have a comparable level of knowledge or expertise. Any such differences may drive respondents to answer questions based on personal opinions or perceptions. Another may be caused by subjectivity, as one person could interpret differently from another based on how he/she understood the ranking presented for each question (*i.e.* from *None* to *More than expected*). Nevertheless, the authors expected that people attending a specialized conference in software measure-

Table 8. Standards used for the definition of measures by practitioners and consultants.

Standard for measurement	% SW companies (n3)	% Consultants (n4)
Internal	66.7	45.5
ISO 9126	33.3	45.5
ISO/IEC 24570:2005 NESMA functional size	0.0	27.3
ISO/IEC 20968:2002 MkII function point	0.0	0.0
ISO/IEC 20926:2003 IFPUG 4.1 unadjusted functional size	41.7	81.8
ISO/IEC 19761:2003 COSMIC-FFP-functional size	25.0	72.7
Other	8.3	36.4

n3 = 14, n4 = 11

Table 9. Level of importance of project management and maintenance measures.

Measures-Level of Importance	Software companies (n3)	
	Proj. Mgmt.	Maintenance
Very Important	33.3%	33.3%
Important	33.3%	16.7%
Normal	16.7%	16.7%
Little Importance	16.7%	33.3%
Not Important	0.0%	0.0%

n3 = 14

ment and filling out the questionnaires would mostly likely have a similar level of knowledge and expertise. The majority of teachers in the sample mentioned that their domain of expertise was software engineering, including software measurement, empirical software engineering and software construction. On the other hand, most of the students indicated that they were enrolled in software engineering programs or specializations. Also, all of them were enrolled in graduate studies, either doctorate candidates or students pursuing a master degree.

In terms of external validity, there are two possible threats. One is the small number of respondents for each type of questionnaire, which does not allow the authors to make generalizations. Another has to do with the fact that the sample was not taken at random. That is, it was deliberately drawn from people attending the IWSM-MENSURA 2010 conference. However, for the type of study presented in this paper, the sampling method is acceptable for the scope of this initial survey. Nevertheless, generalizations about software measurement in education cannot yet be inferred from the results presented in this

paper. A further study has been planned to include a more representative sample by using a Web-based survey.

5. Conclusions and Future Work

Two initial studies have been conducted by researchers at the École de technologie supérieure (ETS) to gain insights into what and how software measurement topics are taught in university curricula: 1) a literature review of publications related to the teaching of software measurement, as summarized in Villavicencio and Abran 2010 [10]; and 2) a survey of teachers, students, practitioners, and consultants on software measurement in an academic environment and in software-related organizations.

As indicated in Section 3, the initial results seem to suggest that specific topics, such as measures for the requirements phase (topic F), and measurement techniques and tools (topic D), are receiving more attention in an academic context, and they confirm the findings reported in Villavicencio and Abran 2010. Specifically, people in academia apparently have a preference for teaching on measurement tools and techniques, as well as for the collection of data in the requirements and programming phases.

Software measurement topics are mostly taught in graduate programs, although such courses are for the most part optional. This finding may explain the commonly held opinion of practitioners and consultants, which is that little or no knowledge of software measurement is brought by students into their companies.

Software measurement is considered as a complex task which demands effort, time and expertise [2-4]. This difficulty is also perceived in the classrooms. In fact, Buglione and Lavazza 2010 [14] reported the difficulties experimented by undergraduate students in understanding the contribution of measures for controlling and monitoring projects. This was evidenced in students without professional experience that were using a procedure called Balancing Multiple Perspectives. This procedure was designed to help project managers choose proper project indicators. One of the explanations for the difficulties just mentioned may be that teachers are mainly using lectures in their courses as indicated in Section 3.1. Giving lectures, however, is not enough for students to reach a deep understanding and higher levels of knowledge [15]. In this respect, the authors maintain that more research is advisable, in order to determine the proper methods for teaching software measurement in an academic environment. These methods must be oriented to facilitate the achievement of higher-order levels of learning by involving students in an active participation in the classrooms.

In future work, we are planning more extensive surveys to provide more comprehensive coverage of this

issue. The goal of these studies is to come up with recommendations as to which software measurement topics should be covered in undergraduate programs.

6. Acknowledgments

The authors thank the IWSM-MENSURA 2010 organizers for allowing us to access and analyze the survey data.

REFERENCES

- [1] J. Iversen and O. Ngwenyama, "Problems in Measuring Effectiveness in Software Process Improvement: A Longitudinal Study of Organizational Change at Danske Data," *International Journal of Information Management*, Vol. 26, No. 1, 2006, pp. 30-43. doi:10.1016/j.ijinfomgt.2005.10.006
- [2] A. Gopal, *et al.*, "Measurement Programs in Software Development: Determinants of Success," *IEEE Transactions on Software Engineering*, Vol. 28, 2002, pp. 863-875. doi:10.1109/TSE.2002.1033226
- [3] C. G. von Wangenheim, *et al.*, "Software Measurement for Small and Medium Enterprises—A Brazilian-German view on Extending the QM Method," *7th International conference on Empirical Assessment in Software Engineering*, Staffordshire, 8-10 April 2003.
- [4] M. Diaz-Ley, *et al.*, "Implementing a Software Measurement Program in Small and Medium Enterprises: A Suitable Framework," *Software, IET*, Vol. 2, No. 5, 2008, pp. 417-436. doi:10.1049/iet-sen:20080026
- [5] M. Díaz-Ley, F. Garcia and M. Piattini, "MIS-PyME Software Measurement Maturity Model-Supporting the Definition of Software Measurement Programs," *Advances in Engineering Software*, Vol. 41, No. 10-11, October-November 2010, pp. 1223-1237.
- [6] J. J. M. Trienekens, *et al.*, "Targets, Drivers and Metrics in Software Process Improvement: Results of a Survey in a Multinational Organization," *Software Quality Journal*, Vol. 15, No. 2, 2007, pp. 135-53. doi:10.1007/s11219-006-9007-y
- [7] C. G. von Wangenheim, *et al.*, "Empirical Evaluation of an Educational Game on Software Measurement," *Empirical Software Engineering*, Vol. 14, No. 4, 2009, pp. 418-452. doi:10.1007/s10664-008-9092-6
- [8] IEEE ACM, "Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering," 2004. <http://sites.computer.org/ccse/SE2004Volume.pdf>
- [9] Integrated Software & Systems Engineering Curriculum (iSSEc) Project, (Graduate Software Engineering 2009 (GSWE2009)), "Curriculum Guidelines for Graduate Degree Programs in Software Engineering," *Stevens Institute of Technology Version 1.0*, 124, 2009. <http://www.gswe2009.org/>
- [10] M. Villavicencio and A. Abran, "Software Measurement in Software Engineering Education: A Comparative Analysis," *International Conferences on Software Measurement IWSM/MetriKon/Mensura 2010*, Stuttgart, 2010, pp. 633-644.

- [11] International Conferences on Software Measurement, *IWSM/MetriKon/Mensura* 2010.
http://www.dasma.org/contray_metrikon_en/html/index.html
- [12] P. Bourque, *et al.*, "SWEBOK Version 3.18," 2008.
<http://www2.computer.org/cms/Computer.org/SWEBOK/MeasurementKA-Draft-Feb2008.pdf>
- [13] C. Reigeluth and A. Carr-Chellman, "Situational Principles of Instruction," In: C. Reigeluth and A. Carr-Chellman, Ed., *Instructional-Design Theories and Models*, Routledge, Taylor & Francis Group, New York, Vol. 3, 2009, pp. 57-71.
- [14] L. Buglione and L. Lavazza, "Suggestions for Improving Measurement Plans: A BMP Application in Italy," *IWSM/MetriKon/Mensura* 2010, Stuttgart, 2010, pp. 361-379.
- [15] J. Biggs and C. Tang, "Teaching for Quality Learning at University," 3rd Edition, Society for Research into Higher Education & Open University Press, Buckingham, 2007.