Measuring the SWEBOK Coverage: An Approach and a Tool

Nicolas Guelfi, Alfredo Capozucca, Benoit Ries

Abstract The definition of a software engineering body of knowledge (SWEBOK) is an important milestone in the history of the software engineering discipline. One of the main questions that might be asked in front of such knowledge definition is: to which extent does my knowledge cover this body of knowledge ? In a more general perspective we can be interested in measuring the coverage of any entity w.r.t. the SWEBOK. It could be a book, a paper, a course, ... In this paper, we present the method we defined to answer such question, the tool we developed and the experiments we did with the lessons learned.

1 Introduction

Understanding what is meant by "software engineering" is still nowadays[1] a not easy task. The answers spectrum starts with classical definitions as the following one:

> The disciplined application of engineering, scientific, and **mathematical principles, methods, and tools** to the economical production of quality software.
> (Watts S. Humphrey in [5])

In order to be more precise on what SE is, the important standardization effort by ISO and IEEE made during the last decade, delivered the software engineering body of knowledge [6]. This body of knowledge defines 15 knowledge areas (KA) decomposed into topics and sub-topics.

Among all the actors that are interested in exploiting the SWEBOK we foresee:

- Education institutions: educating people to become software engineers needs to agree on which notions should be covered. Those institutions thus could use the SWEBOK to either design and maintain their education program [2], or to evaluate to which extent an existing program covers the SWEBOK.
- Industry: in order better handle the recruitment, the management of long life learning, the task allocation, industry professionals could used the SWEBOK coverage as a basis.

N. Guelfi, A. Capozucca, B. Ries
University of Luxembourg
Campus Belval
Avenue de l'Université, 2
L-4365 Esch-sur-Alzette
LUXEMBOURG
E-mail: nicolas.guelfi@uni.lu

[1]and since the well known NATO conference [3]

The problems we are interested in is: what could be an efficient tool-supported method that one could use to specify a targeted or effective coverage of the SWEBOK ?

This paper presents our approach to answer this question by first describing the method, then presenting the tool we developed and finally by illustrating the method and the tool using some case studies we did.

2 The Method

The SWEBOK Coverage Measurement Method that we propose is defined by a process which is described below.

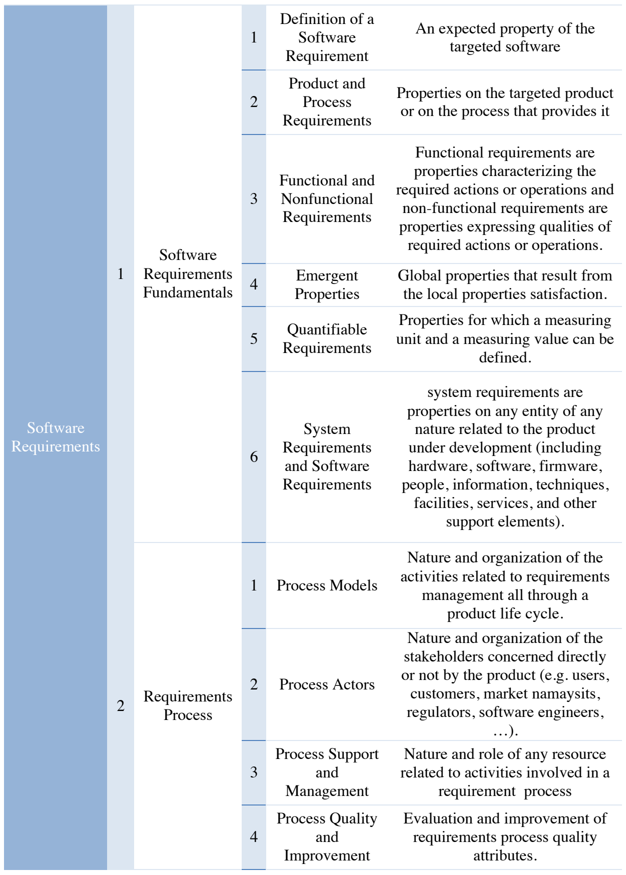
- **Step 1 - Focus Range Selection**: the SWEBOK structure is made of 15 knowledge areas (KA), 100 topics (TP) and 395 sub-topics (STP). It is thus important to determine which knowledge areas and which topics should be considered for the coverage analysis.
- **Step 2 - Coverage Measurement**: for each sub-topic of the selected topics provide the coverage measure values according to the following sub-steps:
  - **Step 2.1 - Sub-topic understanding**: read the synthetic definition provided for the current subtopic in order to have a clear understanding of what knowledge is intended to be represented by the sub-topic name (an extract of these definitions is provided in Figure 1.
  - **Step 2.2 - Focus level**: define the focus level of the current sub-topic. Use the following scale: 0 to indicate that it is not at all the focus, 1 to 3 to indicate the focus intensity.

[2]other education program tools, which by the way are related to the SWEBOK, could be used like: SE2014 Undergraduate Curriculum [1] or GSwE2009 Graduate Software Engineering 2009 [9]

  - **Step 2.3 - Bloom input level**: using the Bloom taxonomy for cognitive levels [2] given in Figure 2, define the level considered to be reached before doing the knowledge acquisition activity targeted (use 0 for the unknowing level if appropriate).
  - **Step 2.4 - Bloom output level**: define the cognitive level targeted to be reached after having done the knowledge acquisition activity.
- **Step 3 - Compute Global Coverage Measures**: use the sub-topics measures provided at step 2 to compute the global coverage measures using an acquisition level scale by cognitive and focus levels as the one provided in Figure 3.

Remarks:

1. The sub-topic synthetic definition is intended to provide a first understanding. If necessary an access to the SWEBOK full text might be useful.
2. If the intent of the coverage is to evaluate the coverage of a software engineering course w.r.t. the SWEBOK then the focus indicates to which extent the sub-topic is targeted by the course as a learning outcome. If the objective is to make a self-evaluation of your own knowledge coverage w.r.t. the SWEBOK, then the focus indicates to which extent your education and professional experience focused on this sub-topic.
3. In case the coverage measure is static (no learning activity) then the input and output levels should be the same.



Software Requirements	1	Software Requirements Fundamentals	1	Definition of a Software Requirement	An expected property of the targeted software
			2	Product and Process Requirements	Properties on the targeted product or on the process that provides it
			3	Functional and Nonfunctional Requirements	Functional requirements are properties characterizing the required actions or operations and non-functional requirements are properties expressing qualities of required actions or operations.
			4	Emergent Properties	Global properties that result from the local properties satisfaction.
			5	Quantifiable Requirements	Properties for which a measuring unit and a measuring value can be defined.
			6	System Requirements and Software Requirements	system requirements are properties on any entity of any nature related to the product under development (including hardware, software, firmware, people, information, techniques, facilities, services, and other support elements).
	2	Requirements Process	1	Process Models	Nature and organization of the activities related to requirements management all through a product life cycle.
			2	Process Actors	Nature and organization of the stakeholders concerned directly or not by the product (e.g. users, customers, market namaysits, regulators, software engineers, ...).
			3	Process Support and Management	Nature and role of any resource related to activities involved in a requirement process
			4	Process Quality and Improvement	Evaluation and improvement of requirements process quality attributes.

Fig. 1 Synthetic description of the sub topic knowledge

1. **Remembering**: *Retrieving, recognizing, and recalling relevant knowledge from long-term memory.*
2. **Understanding**: *Constructing meaning from oral, written, and graphic messages through interpreting, exemplifying, classifying, summarizing, inferring, comparing, and explaining.*
3. **Applying**: *Carrying out or using a procedure through executing, or implementing.*
4. **Analyzing**: *Breaking material into constituent parts, determining how the parts relate to one another and to an overall structure or purpose through differentiating, organizing, and attributing.*
5. **Evaluating**: *Making judgments based on criteria and standards through checking and critiquing.*
6. **Creating**: *Putting elements together to form a coherent or functional whole; reorganizing elements into a new pattern or structure through generating, planning, or producing.*

Fig. 2 Bloom 2001 Cognitive Levels description

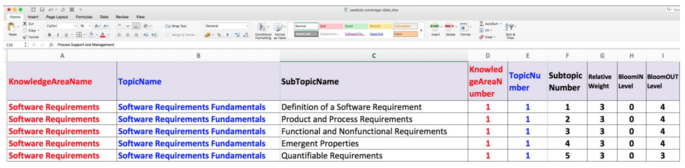
		Focus Levels		
Bloom Levels		1	2	3
Creating	6	80	90	100
Evaluating	5	60	70	80
Analyzing	4	50	55	60
Applying	3	35	40	45
Understanding	2	20	25	30
Remembering	1	5	10	15

Fig. 3 Acquisition Levels Scale (%) by Cognitive and Focus levels

3 The Tool

We engineered a simple tool to support the SWEBOK Coverage Measurement Method. The tool is made of two main components which are described below.

1. **Data Entry Component:** A spreadsheet is providing (as illustrated in Figure 4) that includes: a sheet with the full structured list of the 395 SWEBOK sub-topics to which three columns are added for the focus level, the input and output cognitive levels; a sheet with the synthetic description of the sub-topics.
2. **Data Treatment Component:** A Java program reads the spreadsheet and generates a Prolog data base which is exploited to compute the measurements and to generate the coverage tables and diagrams (as illustrated in Figures 5,6,7 and 8 which are described in the next section).


swebok-coverage-data.xlsx

C10 Process Support and Management

KnowledgeAreaName	TopicName	SubTopicName	KnowledgeAreaNumber	TopicNumber	Subtopic Number	Relative Weight	BloomIN Level	BloomOUT Level
Software Requirements	Software Requirements Fundamentals	Definition of a Software Requirement	1	1	1	3	0	4
Software Requirements	Software Requirements Fundamentals	Product and Process Requirements	1	1	2	3	0	4
Software Requirements	Software Requirements Fundamentals	Functional and Nonfunctional Requirements	1	1	3	3	0	4
Software Requirements	Software Requirements Fundamentals	Emergent Properties	1	1	4	3	0	4
Software Requirements	Software Requirements Fundamentals	Quantifiable Requirements	1	1	5	3	0	3

Fig. 4 Spreadscheet for SWEBOK measures entry

4 Experiments

4.1 Coverage for Software Engineering Project Courses

One of our main targets are academic courses at bachelor or master level that are dedicated to the practical learning of software engineering activities. We have made the coverage analysis for two institutions that offer a software engineering project course:

– **1 - UL**: Bachelor in Informatics - Software Engineering Project and Software Engineering II Courses ([10]).
– **2 - CMU Silicon Valley**: Master in Software Engineering - Foundations of Software Engineering Course, US ([7]).

The coverage tables generated by our tool using a neutral acquisition scale are given in Figures 5 and 6. The SWEBOK knowledge areas are ranged by decreasing coverage order. Those tables allow to easily determine which are the KAs which are the focus of the respective courses (e.g. Software Requirements and Software Engineering Models and Methods for University of Luxembourg; and Software Design and Software Testing for CMU Silicon Valley). It can also be determined which are the KAs poorly addressed (e.g. Software Engineering Economics and Software Configuration Management for University of Luxembourg; and Engineering Foundations and Software Engineering Economics for CMU Silicon Valley).

Nb	Knowledge Area	Cov. (%)
1	Software Requirements	80
9	Software Engineering Models and Methods	75
7	Software Engineering Management	67
11	Software Engineering Professional Practice	63
2	Software Design	46
3	Software Construction	39
8	Software Engineering Process	33
4	Software Testing	32
5	Software Maintenance	28
14	Mathematical Foundations	19
15	Engineering Foundations	18
10	Software Quality	17
13	Computing Foundations	11
12	Software Engineering Economics	8
6	Software Configuration Management	0

Fig. 5 Coverage table for **1 - UL - (Average is 36%)**

Nb	Knowledge Area	Cov. (%)
2	Software Design	89
4	Software Testing	79
3	Software Construction	75
11	Software Engineering Professional Practice	68
10	Software Quality	67
5	Software Maintenance	56
7	Software Engineering Management	54
1	Software Requirements	50
6	Software Configuration Management	44
9	Software Engineering Models and Methods	44
13	Computing Foundations	37
8	Software Engineering Process	33
14	Mathematical Foundations	29
15	Engineering Foundations	18
12	Software Engineering Economics	17

Fig. 6 Coverage table for **2 - CMU Silicon Valley - (Average is 51%)**

4.2 Compartive Coverage for Bachelors in Computer Science

We have also applied our approach to full academic education programs for bachelors in Computer Science. As an example, we provide below the comparative coverage obtained without Bloom levels consideration for the three following bachelors:

– **1 - USC**: Bachelor in Computer science, California State University, US ([4]).
– **2 - MSU**: Bachelor in Computer science, Michigan State University, US ([7]).
– **3 - ULB**: Bachelor in Computer science, Université Libre de Bruxelles, US ([11]).

KA	Name	USC	MSU	ULB
1	Software Requirements	73	70	33
2	Software Design	66	54	17
3	Software Construction	64	67	36
4	Software Testing	32	42	5
5	Software Maintenance	17	0	0
6	Software Configuration Management	17	0	6
7	Software Engineering Management	75	71	58
8	Software Engineering Process	40	33	33
9	Software Engineering Models and Methods	56	100	31
10	Software Quality	8	42	50
11	Software Engineering Professional Practice	84	79	58
12	Software Engineering Economics	8	60	21
13	Computing Foundations	65	79	77
14	Mathematical Foundations	75	69	63
15	Engineering Foundations	29	65	41

Fig. 7 Bachelors Coverage Comparison (%)

The generated table or diagram by our tool, illustrated in Figures 7 and 8 allow to have a first understanding of each program content w.r.t. software engineering. They also allow to have a comparative analysis of the programs w.r.t. each others in terms of SWEBOK coverage[3].

[3]We provide here only global coverage measures at knowledge areas level but of course one can choose to observe at topic or even sub-topic levels).

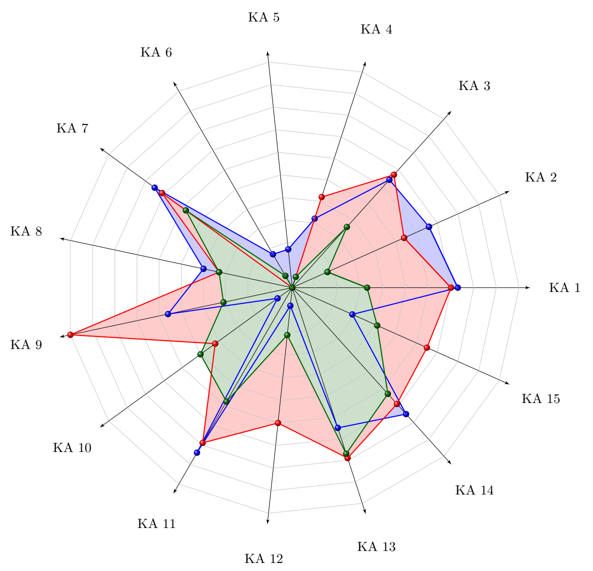


Fig. 8 Coverage Comparison table

5 Lessons Learned and Conclusion

The development of the method and its tool together with the experiments made allowed us to obtain some interesting outcome.
For what concerns the method, the experiments made shown the following:

– Defining the measures is of course the key point. In our study we ensure consistency by doing ourselves all the measures (except for the software engineering course at CMU). In any case those values represent the knowledge of the evaluator of the SWEBOK but also of the observed entities (a program, a individual, . . .). It is thus very important to understand who are the evaluators and with which inputs they evaluated the measurement values.
– The need to access the full SWEBOK book increases the complexity to proceed to the coverage analysis. This is why the method is applicable only if the user can provide the measures only using the knowledge area, topic and sub-topic terms completed with the synthetic description provided.
– Having a configurable acquisition scale by focus and cognitive levels is very important since this allow to represent the individual coverage understanding by the users of the method.

For what concerns the tool, the experiments made shown the following:

– many operational faults are possible in entering the data in the spreadsheet.
– it takes from 60 to 90 minutes to fill all the values for the 395 topics (depending on if you use, or not, multiple fill for topics or even knowledge areas when you think that they should have the same measures).

To conclude, we have developed an interesting method and tool that goes far beyond the one followed and presented in [8]. Our tool needs to be redesigned and made available through a web interface to the community. This is what we expect to do in the coming months depending on the support we will find.
Concerning the experiments, we intend to start some large scale experiments:

– an experiment allowing to have a detailed analysis of the SWEBOK coverage at bachelor levels by continents.
– an experiment to evaluate the actual knowledge acquisition w.r.t. SWEBOK sub-topics. Some software engineering courses will be selected and students will use the method and the tool to evaluate their SWEBOK coverage before and after the course.
– an experiment to evaluate the actual knowledge w.r.t. SWEBOK sub-topics by academics in charge of software engineering courses.

Acknowledgements The authors would like to thank especially Dr. Cécile Péraire from CMU Silicon Valley for having applied the method and used the tool for what concerns her Foundations of Software Engineering Course.

References

1. ACM/IEEE (2015) Software Engineering 2014 - Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. ACM, New York, NY, USA, URL https://www.acm.org/education/SE2014-20150223_draft.pdf,https://www.acm.org/education/curricula-recommendations
2. Anderson LW, Krathwohl DR, Bloom BS (2001) A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives. Allyn & Bacon
3. Bauer FL (1971) Software engineering. In: 1. Foundations and systems., International Federation for Information Processing: IFIP congress series, pp 530–538
4. California State University (2015) Bachelor in computer science. http://catalogue.usc.edu/preview_program.php?catoid=2&poid=1486, accessed: 2015-06-05
5. Humphrey WS (1989) Managing the software process. Reading, Mass: AddisonWesley
6. ISO/IEC (2014) Software Engineering – Guide to the Software Engineering Body of Knowledge (SWEBOK). International Organization for Standardization, iSO-IEC TR 19759-2014
7. Michigan State University (2015) Bachelor in computer science. http://www.cse.msu.edu/Resources/CSECourseInformation.php, accessed: 2015-06-05
8. Pyster A, Turner R, Henry D, Lasfer K, Bernstein L (2009) Master's degrees in software engineering: An analysis of 28 university programs. IEEE software 26(5):94–101
9. Pyster A, et al (2009) Graduate software engineering 2009 (gswe2009) curriculum guidelines for graduate degree programs in software engineering. Stevens Institute of Technology
10. University of Luxembourg (2016) Bachelor in informatics - software engineering project and software engineering ii courses. http://www.cse.msu.edu/Resources/CSECourseInformation.php, accessed: 2016-05-01
11. Université Libre de Bruxelles (2015) Bachelor in computer science. http://banssbfr.ulb.ac.be/PROD_frFR/bzscrse.p_disp_prog_detail?term_in=201415&prog_in=BA-INFO&lang=FRENCH#cursus_section, accessed: 2015-06-05
=2