Assessment Of Course Contents Of Software Engineering Course (Undergraduate) At The Universities In Kingdom Of Saudi Arabia

Dr. Raza Ur Rehman Qazi

Abstract: In this study an assessment has been made of the Software Engineering course being taught at the undergraduate level in the Universities at the Kingdom of Saudi Arabia. The course being taught at the department of Computer science, college of computer and Information Sciences, Imam Muhammad Ibn Saud Islamic University has been selected as sample. For this study the software engineering course (CS290) offered at the department of computer science has been considered. For the course contents assessment the standard recommended by the software engineering body of knowledge (SWEBOK) has been used as a bench mark. As a result of this study some major gaps were found in the SWEBOK and the CS290 course. In some topics the gap is wider where the SWEBOK is proposing application level while the CS290 level is knowledge. Detailed are presented in analysis and result section. Based on the outcome of this study, recommendations have been made for improvement in the course contents to fill the gap.

Keywords: Software Engineering Body of Knowledge (SWEBOK), Software Engineering Education Knowledge (SEEK), Association of Computing Machinery (ACM), Computer Society of the Institute of Electrical and Electronic Engineers (IEEE-CS), Software Engineering, Computational thinking

Introduction

In this day and age software has a role in almost every aspect of our lives. The increase in the use of software focuses attention on both the technology itself, and on the effect of that technology on society. Indeed, software projects are seen as being central to the growth of the information economy. To support this growth, it is important that suitable methods of software teaching, training, and development are adopted and those developing software obtain the relevant skills. Many in the Information Technology (IT) industry carry the position title of "Software Engineer", but professional recognition has been slow. The motivation to conduct this study came from the results of a study conducted by Stephanie Ludil and James Collofello. This results of this study suggests that there is a gap between the course contents being taught at academia and what is actually practiced in the industry [2]. Keeping in view these findings (gap between industry practices and course taught at academia) the author felt that it will be useful for the academia in KSA to find out the gap if it exists. And if so to suggests ways to improve it. Guidelines from Software Engineering Body of Knowledge (SWEBOK) [1] will be used as the basic bench mark for assessing the course contents.

Terminology

The Software Engineering Body of Knowledge Project (SWEBOK) is a joint effort by the IEEE Computer Society and the Association for Computing Machinery (ACM) to develop a guide to the subset of generally accepted knowledge that defines the Software Engineering profession.

- *Dr. Raza Ur Rehman Qazi Assistant Professor, department of computer science Imam Muhammed Ibn Saud Islamic University, Saudi Arabia*

Bloom's taxonomy: A way of mapping of topics within the SWEBOK Knowledge Areas.
Software engineering: A discipline that is concerned with all aspects of software production from the early stages of inception and specification to the maintenance of the system when it has gone into use.
Computational thinking as defined by Jeannette Wing[7] is a way of solving problems, designing systems, and understanding human behavior by drawing on concepts that are fundamental to computer science.

Assumptions

For this study the following assumptions when specifying the taxonomy levels.

- This study is for a general software engineer and not for specialized field in software engineering like for example software testing etc.
- The topics of software engineering management and software configuration management are not included because these topics are being covered by another course, CS494-Software Engineering Management offered in the department of computer science at the Imam Muhammad Ibn Saud Islamic University.
- This study is considered for the new beginners in the field of software engineering. It is assumed that this study if for a software engineer with 1-2 years of industry experience.
- The software engineers are assigned relatively few management duties, or at least not for major endeavors. "Management-related topics".
- The taxonomy levels are lower for "early life cycle topics" such as those related to software requirements than for more technically-oriented topics such as those within software design, software construction or software testing.

Bloom's Taxonomy of the Cognitive Domain proposed in 1956 contains six levels. The following table presents these levels and keywords often associated with each level.

Bloom's Taxonomy (SWEBOK, 2004)

Bloom's Taxonomy Level	Associated Keywords
Knowledge: Recall data	Defines, describes, identifies, knows, labels, lists, matches, names, outlines, recalls, recognizes, reproduces, selects, states
Comprehension: Understand the meaning, translation, interpolation, and interpretation of instructions and problems; state a problem in one's own words.	Comprehends, converts, defends, distinguishes, estimates, explains, extends, generalizes, gives examples, infers, interprets, paraphrases, predicts, rewrites, summarizes, translates
Application: Use a concept in a new situation or use an abstraction unprompted; apply what was learned in the classroom to novel situations in the workplace	Applies, changes, computes, constructs, demonstrates, discovers, manipulates, modifies, operates, predicts, prepares, produces, relates, shows, solves, uses
Analysis: Separate material or concepts into component parts so that its organizational structure may be understood; distinguish between facts and inferences	Analyzes, breaks down, compares, contrasts, diagrams, deconstructs, differentiates, discriminates, distinguishes, identifies, illustrates, infers, outlines, relates, selects, separates
Synthesis: Build a structure or pattern from diverse elements; put parts together to form a whole, with emphasis on creating a new meaning or structure	Categorizes, combines, compiles, composes, creates, devises, designs, explains, generates, modifies, organizes, plans, rearranges, reconstructs, relates, reorganizes, revises, rewrites, summarizes, tells, writes
Evaluation: Make judgments about the value of ideas or materials	Appraises, compares, concludes, contrasts, criticizes, critiques, defends, describes, discriminates, evaluates, explains, interprets, justifies, relates, summarizes, supports

Data Analysis and Results

Bloom's Taxonomy Matrix (CS290 against SWEBOK) report

For this study the taxonomy levels for the CS290 course are proposed by the faculty members who have taught CS290 course. The faculty members include, Dr. Rouf, Dr. Hedi Haddad, Dr. Alawairdhi and myself. Faculty members from 5 other universities at KSA were also consulted in order to make the result generalized to the Software engineering courses offered at other universities in KSA. In the software requirements case we found that most of the levels proposed by the SWEBOK were not met by the CS290 course taught at the department of Computer Science of the IMAM's University. Some major gaps were found in the SWEBOK and the CS290 course. According to the SWEBOK the topic of emergent properties is required to be covered to the level of comprehension but in the CS290 it is not covered at all. Similar is the case with the topics of process actors, requirement sources, model validation, and measuring requirements. In some of the topics the SWEBOK suggests comprehension but the CS290 is providing knowledge. In some topics the gap is wider where the SWEBOK is proposing application level while the CS290 level is knowledge. These gapes can be seen in table-1, which includes the topics of requirements classifications, software requirements specifications, and change management. In table-2 the software design skills are evaluated. The comparison table indicates a big gap between the standard level proposed by SWEBOK and the CS290. Most of the important topics are not covered by the CS290. The topics highlighted in red color are not covered by the CS290 course. Software construction topics are compared in table-3. The topics of minimizing complexity, constructing verification, and construction design are not covered in the CS290 course. The gap is also wide between the SWEBOK recommended standard and the topics CS290 in the areas of anticipating change, standards in construction, construction planning, construction measurement, construction testing, construction quality, and integration. The gap is relatively small between the two (SWEBOK and CS290) in coding (i.e. analysis and application). The software testing analysis (table-4) indicates that in CS290 most of the important topics recommended by SWEBOK are not covered. The topics not covered by CS290 course includes Relationships of testing to other activities, The target of the tests, Based on tester's intuition and experience, Specification-based, Fault-based, Usage-based, Based on nature of application, selecting and combining techniques, Management concerns. The analysis further indicated that the topics of software maintenance (table-5), software engineering tools (table-7) and software quality (table-8) are not covered by the CS290 course. This observation shows that the CS290 course contents need a detailed review in order to make it according to the standard recommended by the SWEBOK, which is followed worldwide.

Table 1: SOFTWARE REQUIREMENTS

Software Engineering topic	Taxonomy (SWEBOK)	CS290
1. Software requirements fundamentals		
Definition of software requirement	C	K
Product and process requirements	C	K
Functional and non-functional requirements	C	K
Emergent properties	C	
Quantifiable requirements	C	K
System requirements and software requirements	C	K
2. Requirements process		
Process models	C	K
Process actors	C	
Process support and management	C	K
Process quality and improvement	C	K
3. Requirements elicitation		
Requirements sources	C	
4. Requirements analysis		
Requirements classification	AP	K
Conceptual modeling	AN	AN
Architectural design and requirements allocation	AN	AN
Requirements negotiation	AP	AP
5. Requirements specification		
System definition document	C	K
System requirements specification	C	K
Software requirements specification	AP	K
6. Requirements validation		
Requirements reviews	AP	AN
Prototyping	AP	AP
Model validation	C	
Acceptance tests	AP	AP
7. Practical considerations		
Iterative nature of requirements process	C	K
Change management	AP	K
Requirements tracing	AP	AP
Measuring requirements	AP	

K: Knowledge, C: Comprehension, AP: Application, AN: Analysis, E: Evaluation, S: Synthesis (SWEBOK, 2004)

Table 2: SOFTWARE DESIGN

Software Engineering topic	Taxonomy (SWEBOK)	CS290
1. Software design fundamentals		
General design concepts	C	C
Context of software design	C	K
Software design process	C	AP
Enabling techniques	C	
2. Key issues in software design		
Concurrency	AP	
Control and handling of events	AP	
Distribution of components	AP	
Error and exception handling and fault tolerance	AP	AP
Interaction and presentation	AP	
Data persistence	AP	
3. Software structure and architecture		
Architectural structures and viewpoints	AP	
Architectural styles (macro architectural patterns)	AN	
Design patterns (micro architectural patterns)	AN	
Families of programs and frameworks	C	
4. Software design quality analysis and evaluation		

Quality attributes	C	C
Quality analysis and evaluation techniques	AN	AN
Measures	C	
5. Software design notations		
Structural descriptions (static)	AP	AP
Behavioral descriptions (dynamic)	AP	AP
6. Software design strategies and methods		
General strategies	AN	K
Function-oriented (structured) design	AP	AP
Object-oriented design	AN	AN
Data-structure centered design	C	
Component-based design (CBD)	C	
Other methods	C	

Table 3: SOFTWARE CONSTRUCTION

Software Engineering topic	Taxonomy (SWEBOK)	CS290
1. Software construction fundamentals		
Minimizing complexity	AN	
Anticipating change	AN	K
Constructing for verification	AN	
Standards in construction	AN	K
2. Managing construction		
Construction methods	C	C
Construction planning	AP	C
Construction measurement	AP	K
3. Practical considerations		
Construction design	AN	
Construction languages	AP	AP
Coding	AN	AP
Construction testing	AP	K
Construction quality	AN	K
Integration	AP	K

Table 4: SOFTWARE TESTING

1. Software testing fundamentals	Taxonomy (SWEBOK)	CS290
Testing-related terminology	C	K
Key issues	AP	K
Relationships of testing to other activities	C	
2. Test levels		
The target of the tests	AP	
Objectives of testing	AP	K
3. Test techniques		
Based on tester's intuition and experience	AP	
Specification-based	AP	
Code-based	AP	K
Fault-based	AP	
Usage-based	AP	
Based on nature of application	AP	
Selecting and combining techniques	AP	
4. Test-related measures		
Evaluation of the program under test	AN	K
Evaluation of the tests performed	AN	K
5. Test process		
Management concerns	C	
Test activities	AP	K

Table 5: SOFTWARE MANITENANCE

Software Engineering topic	Taxonomy(SWEBOK)	CS290
1. Software maintenance fundamentals		
Definitions and terminology	C	K
Nature of maintenance	C	K
Need for maintenance	C	
Majority of maintenance costs	C	
Evolution of software	C	K
Categories of maintenance	AP	
2. Key issues in software maintenance		
Technical		
Limited understanding	C	
Testing	AP	
Impact analysis	AN	
Maintainability	AN	
Management issues		
Alignment with organizational issues	C	AP
Staffing	C	AP
Process issues	C	
Organizational	C	
Maintenance cost estimation		
Cost estimation	AP	AN
Parametric models	C	
Experience	AP	
Software maintenance measurement	AP	
3. Maintenance process		
Maintenance process models	C	
Maintenance activities		
Unique activities	AP	
Supporting activities	AP	
4. Techniques for maintenance		
Program comprehension	AN	
Reengineering	C	
Reverse engineering	C	K

Table 5: SOFTWARE ENGINEERING PROCESS

Software Engineering topic	Taxonomy (SWEBOK)	CS290
1. Process implementation and change		
Process infrastructure	C	
Software engineering process group	C	
Experience factory	C	
Activities	AP	
Models for process implementation and change	K	K
Practical considerations	C	K
2. Process definition		
Life cycle models	AP	C
Software life cycle processes	C	C
Notations for process definitions	C	K
Process adaptation	C	K
Automation	C	K
3. Process assessment		
Process assessment models	C	
Process assessment methods	C	
4. Product and process measurement		
Software process measurement	AP	

Software product measurement	AP	AP
Size measurement	AP	AP
Structure measurement	AP	AP
Quality measurement	AP	AP
Quality of measurement results	AN	
Software information models		
Model building	AP	
Model implementation	AP	
Measurement techniques		
Analytical techniques	AP	
Benchmarking techniques	C	

Table 7: SOFTWARE ENGINEERING TOOLS

Software Engineering topic	Taxonomy (SWEBOK)	CS290
1. Software tools		
Software requirements tools	AP	
Software design tools	AP	
Software construction tools	AP	
Software testing tools	AP	
Software maintenance tools	AP	
Software engineering process tools	AP	
Software quality tools	AP	
Software configuration management tools	AP	
Software engineering management tools	AP	
Miscellaneous tool issues	AP	
2. Software engineering methods		
Heuristic methods	AP	
Formal methods and notations	C	

Table 6: SOFTWARE QUALITY

Software Engineering topic	Taxonomy (SWEBOK)	CS290
1. Software quality fundamentals		
Software engineering culture and ethics	AN	
Value and costs of quality AN	AN	K
Quality models and characteristics		
Software process quality AN	AN	
Software product quality AN	AN	
Quality improvement AP	AP	
2. Software quality management processes		
Software quality assurance AP	AP	
Verification and validation AP	AP	
Reviews and audits		
Inspections AP	AP	
Peer reviews AP	AP	
Walkthroughs AP	AP	
Testing AP	AP	K
Audits C	C	
3. Practical considerations		
Application quality requirements		
Criticality of systems C	C	
Dependability C	C	
Integrity levels of software C	C	
Defect characterization AP	AP	

Software quality management techniques		
Static techniques AP	AP	
People-intensive techniques AP	AP	
Analytic techniques AP	AP	
Dynamic techniques AP	AP	
Software quality measurement AP	AP	

Conclusion and future recommendations

The purpose of this study was to assess and analyze the contents of a software engineering (CS290) course offered at the department of computer science at the college of computer and information sciences at the Imam Muhammed Ibn Saud Islamic University, KSA. In the process of assessment the course contents of the CS290 course were examined against the standard recommended by the Software Engineering Body of Knowledge (SWEBOK). After applying the bloom's taxonomy matrix, it was found that most of the course content levels proposed by the SWEBOK were not met by the CS290 course taught at the department of Computer Science of Imam's University. Some major gaps were found between the course contents recommended by the SWEBOK and the contents CS290 course syllabus. Some of the most notable gapes were found in the topics of requirement sources, model validation, measuring requirements, process actors, and emergent properties (table1-table8). According to the SWEBOK the topic of emergent properties is required to be covered to the level of comprehension but in the CS290 it is not covered at all. Similar is the case with the topics of process actors, requirement sources, model validation, and measuring requirements. In some of the topics the SWEBOK suggests comprehension but the CS290 is providing knowledge. In some topics the gap is wider where the SWEBOK is proposing application level while the CS290 level is knowledge. Based on the results of this study it is highly recommended to review the CS290 course in light of recommendation made by SWEBOK.

References

[1] Abran, A., and Moorc, J. (cds.) "Guide to the Software Engineering Body of Knowledge", A Stone Man Version (version .6). http://www.swebok.org/ironrnan

[2] Stephanie Ludil and James Collofello "an analysis of the gap between the knowledge and skills learned in academic software engineering course projects and those required in real projects" October 2001

[3] Bourquc, P., Dupuis, R., ct al. "The Guide to the Software Engineering Body of Knowledge". IEEE Softwure. Vol. 16, No 6, November-December 1999, pp. 35-44.

[4] IEEE Computer Society, Guide to the SWEBOK http://www.swebok.org/documents/

[5] IEEE Computer Society, Guide to the SWEBOK http://www.swebok.org/overview/

[6] Sommerville, I. 2007. Software Engineering 8th Ed. Boston: Addison Wesley.

[7] United States Department of Labor: Bureau of Statistics. Occupational Outlook Handbook.

[8] Wing, J.M. "Computational Thinking." Communications of the ACM. (49) 3:33 – 35. http://www.cs.cmu.edu/afs/cs/usr/wing/www/publications/Wing06.pdf

[9] The Tuskegee University Bulletin Courses and Programs 2004 – 2006. http://www.tuskegee.edu.

[10] Lester, C. CSCI 430- Software Engineering Course Syllabus. 2008.