Research Journal of Applied Sciences, Engineering and Technology 13(2): 98-106, 2016
DOI:10.19026/rjaset.13.2920
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: March 19, 2014 Accepted: April 22, 2014 Published: July 15, 2016

Research Article

Assessment of Student's Perception and Teaching Methodology of Software Engineering Course (Undergraduate) at Universities in Kingdom of Saudi Arabia

Mohammed Alawairdhi
Al-Imam Muhammad ibn Saud Islamic University, Riyadh, Saudi Arabia

Abstract: In this research an attempt has been made to assess the students' perceptions of learning the software engineering course and the teaching method being used to deliver the Software Engineering course at the undergraduate level at the Department of Computer science, college of computer and Information Sciences, Imam Muhammed Ibn Saud Islamic University, Riyadh, KSA. For this study we are considering the software Engineering Course (CS290) offered at the Department of Computer science, college of computer and Information Sciences, Imam Muhammed Ibn Saud Islamic University, Riyadh, KSA. The teaching method adopted to teach the CS290 course has been assessed in order to find out if the currently adopted approach matches the lately discovered 4 teaching methods for the delivery of Software Engineering course at undergraduate level. The four teaching methods (Formal authority, Personal model, Facilitator and Delegator) described by Grasha (1994) will be considered as bench mark. The students' perception about the course contents and its importance in the industry will be assessed based on the survey. The teaching method is assessed from the teachers who are teaching this course in the last 5 years. Based on the outcome of this study, recommendations have been made for improvement.

Keywords: Association of Computing Machinery (ACM), Computer Society of the Institute of Electrical and Electronic Engineers (IEEE-CS), Computational thinking, Software Engineering Body of Knowledge (SWEBOK), collaborative learning, Software Engineering Education Knowledge (SEEK), student-centered and teacher-centered approaches, software engineering, Service-learning

INTRODUCTION

According to the Information Society of KSA, the government of KSA is committed to invest in IT. This report states that "The Government implemented a multi-stage plan for restructuring the ICT sector with the objectives of encouraging effective competition, attracting local and foreign investment, as well as protecting public interest and consumer and stakeholder rights. The state-run telecommunications organization was incorporated in 1989 as the Saudi Telecom Company (STC) and was partially privatized in 2003. The Communications and Information Technology Commission (CITC) was established in 2001 as the regulatory authority with legal standing and financial and administrative independence. The Commission Statutes ("Telecommunications Act", the "Bylaw" and the "Ordinance" and the "Rules of procedures") were also enacted, which can be found on the CITC website. The Government has taken a number of steps to liberalize the market and create a positive regulatory framework to encourage investment and promote growth of the ICT market. By 2004, competition was introduced in the mobile, data and VSAT telecom areas. Competition in the fixed services and more mobile market liberalization was introduced by issuing new licenses in 2007". A number of initiatives and policies have been developed to stimulate spread and usage of the Internet. As a result, ICT services have been improving in terms of scope, quality and lower prices to the consumers. Saudi Arabia acceded to the World Trade Organization (WTO) as its 149th member in December 2005. As part of the WTO commitment, Saudi Arabia is committed to liberalize its ICT sector in accordance with, the General Agreement of Trade in Services (GATS), the Agreement on Basic Telecommunications (ABT) and the Reference Paper. Saudi Arabia also commits to extend non-preferential treatment to all other WTO members, be highly transparent in its regulations and provide full market access for almost all its telecommunications services."

To develop quality software needs a lot of efforts from all concerned bodies like organizations, software engineers and academia (Sommerville, 2007). In order to improve the learning curve of fresh software graduates it is very important that academia is constantly in touch with the industry. It is the responsibility of the IT departments to educate new software engineers in a way that is consistent to the current trends in the software engineering industry. The

key guide for those who are involved with software engineering is the Software Engineering Body of Knowledge (SWEBOK). This guide is the collection of best practices of software engineering in the industry (McPherson, 2005).

The Software Engineering Body of Knowledge Project (SWEBOK) is a collection of generally accepted knowledge of the software engineering profession. This guide was the result of a joint project by the IEEE Computer Society and the Association for Computing Machinery (ACM). The purpose of this guide is not to define the body of knowledge or to dictate the curricula for university programs (United States Department of Labor, 2008). However, this guide can be used to assist in the development of curricula and accreditation criteria. The overall goals of the Guide to the Software Engineering Body of Knowledge are to:

- Characterize the contents of the Software Engineering Body of Knowledge
- Provide topical access to the Software Engineering Body of Knowledge
- Promote a consistent view of software engineering worldwide
- Clarify the place of and set the boundary of software engineering with respect to other disciplines such as Computer Science, Project Management, Computer Engineering and Mathematics
- Provide a foundation for curriculum development and individual certification and licensing material (IEEE Computer Society, 2006)

SWEBOK consists of three phases: Strawman, Stoneman and Ironman. The Strawman phase has been completed and has resulted in a guide presenting the Knowledge Areas and Related Disciplines (Felder and Brent, 1996). The objective of the Stoneman version of the guide is to organize the body of knowledge into Knowledge Areas, a list of topics relevant to the materials for each Knowledge Area and a list of Related Disciplines (Bourque *et al*., 1999). The list of 10 core knowledge areas (Meyers and Jones, 1993) and the topics that comprise them are given below:

- Software configuration management
- Software construction
- Software design
- Software engineering infrastructure
- Software engineering management
- Software engineering process
- Software evaluation and maintenance
- Software quality analysis
- Software requirements analysis
- Software testing

The lronman work facilitates experimentation and trial usage of the guide, the promotion of the guide and the development of "performance norms" for professionals (Abran and Moore, 2009). This guide was the result of the continuing collaboration of individuals from industry, academia and standard setting bodies from all over the world (The Tuskegee University Bulletin, 2004).

MATERIALS AND METHODS

Four styles of teaching:
Formal authority: Instructor-centered approach where the instructor provides and controls the flow of content for the course.

Demonstrator/personal model: Instructor centered approach where the instructor demonstrates the skills that students are expected to learn.

Facilitator: Student-centered approach where the instructor acts as a facilitator and the responsibility is placed on the student to achieve results for various tasks.

Delegator: Student-centered approach where the instructor delegates and places the control and the responsibility for learning on the students and/or groups of students.

Terminology:
The Software Engineering Body of Knowledge Project (SWEBOK): 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.

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 (Gokhale, 1995).

Computational thinking: As defined by Wing (2006) is a way of solving problems, designing systems and understanding human behavior by drawing on concepts that are fundamental to computer science.

Collaborative learning: Defined as the grouping and/or pairing of students for the purpose of achieving an academic goal (Grasha, 1994).

Informal learning groups: Are ad hoc temporary grouping of students within a single class period (Johnson and Johnson, 1994).

Formal learning groups: Are teams established to complete a specific task. These groups may complete their work in a single class period or over the course of several weeks.

Computing Curriculum-Software Engineering (CCSE): Provide guidance to academic institutions and accreditation agencies about what should constitute an undergraduate software engineering education (Wang and Lin, 2007).

Service-learning: Defined as a method of teaching through which students apply their academic skills and knowledge to address real-life needs in their own communities (Wing, 2006).

Methodology:

- A survey is used to collect the student's perception about the importance of the CS290 course.
- Teachers who have taught the selected course will be asked (questionnaire) about their teaching style.
- Based on the result it will be find out what method has been used by the teacher.
- The findings will be compared against the 4 well known of teaching methods and recommendations will be made accordingly.

Tools used:

- Questionnaire
- Excel Spread sheet
- SPSS for data analysis

Assumptions: For this study we are considering the following assumptions (almost the same proposed by SWEBOK) 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.
- In this study we are not considering the topics of software engineering management and software configuration management because these topics are being covered by another course, CS494-Software Engineering Management offered in the department of computer science at the Imam Muhammed Ibn Saud University.
- This study is considering the new beginners in the field of software engineering. We assume that a software engineer with 1-2 years of experience of industry.
- The software engineer are assigned relatively few management duties, or at least not for major endeavors. "Management-related topics".

RESULTS AND DISCUSSION

Survey analysis report: The data analysis shown in the following graphs suggests that overall the student's perception of readiness for employment by taking the

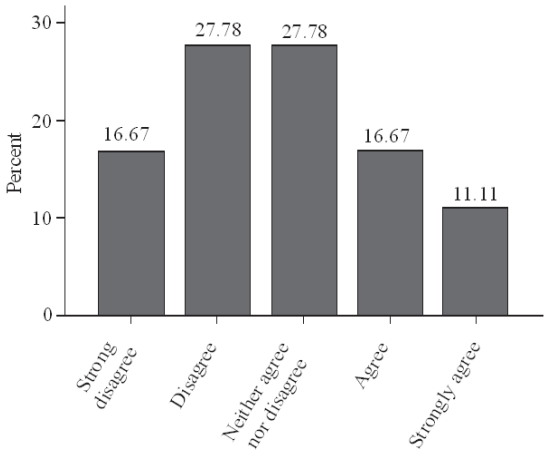

Fig. 1: I felt confident in starting work

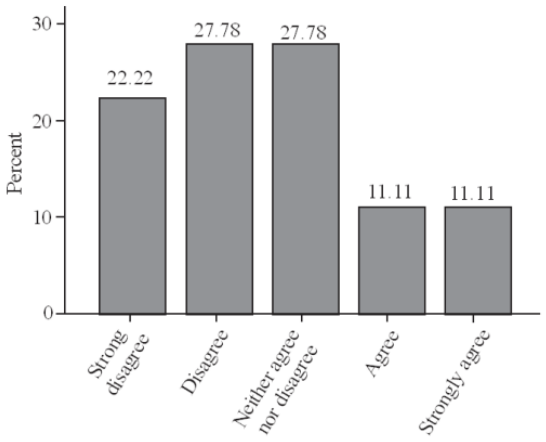

Fig. 2: The knowledge and skills from the software engineering course helped me at my job

software engineering course is not satisfactory. Significant numbers of students think that they are not confident to start working as a software engineer or similar role. The data indicates that the software engineering course (CS290) didn't fully prepare them for the job of software engineer. Let's go through each question asked to the survey and analyze them to get a deeper understanding of the factors affecting the student's perceptions.

Figure 1 shows that most students (75%) think that they are not ready to accept the responsibility of a software engineer.

Most of the students (78%) are also of the opinion that they didn't get the knowledge and the skills that are required for the position of a beginner software engineer (Fig. 2).

Similarly, Fig. 3 suggests that 70% students are either not satisfied or neutral when they are asked if the employer valued their knowledge and skills gained from the CS290 course.

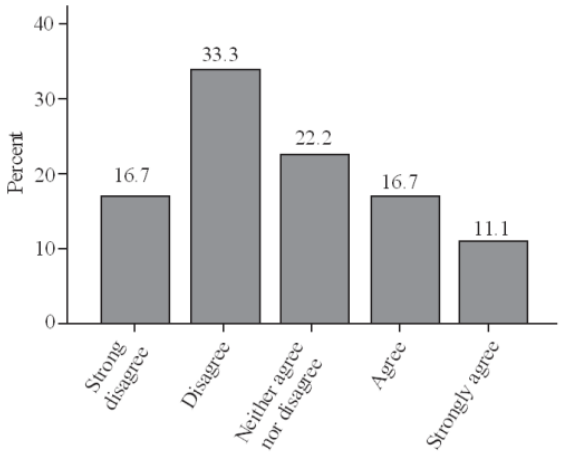

Fig. 3: In employing me. I felt that my employer valued highly the knowledge I had gained in my software engineering course

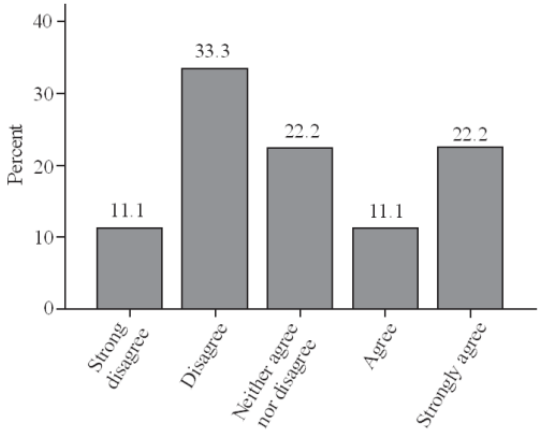

Fig. 4: After commencing work. My employer quickly gave me added responsibilities. Realizing I was capable of working at a higher level

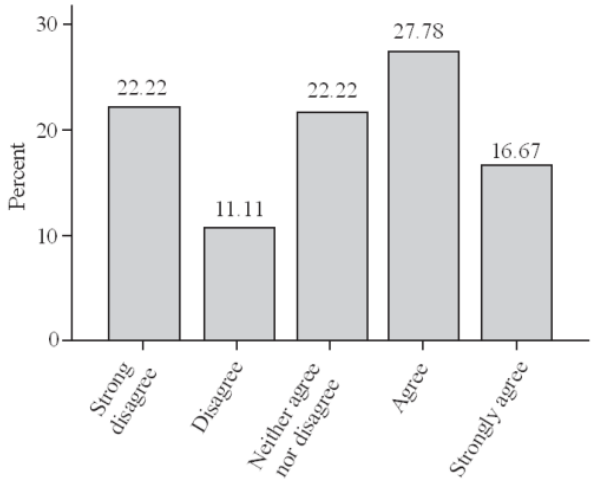

Fig. 5: I have gained the project management skills during my studies.

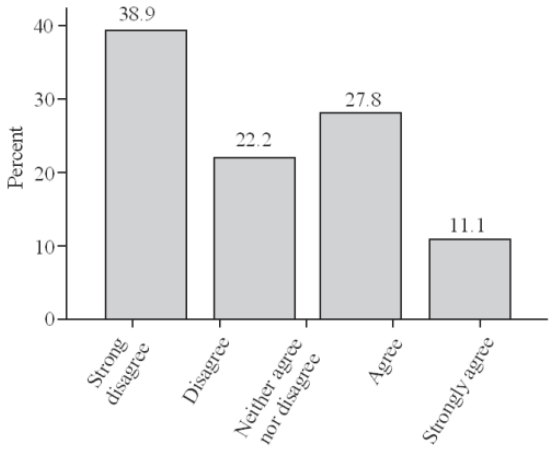

Fig. 6: I have gained the quality assurance skills during my studies

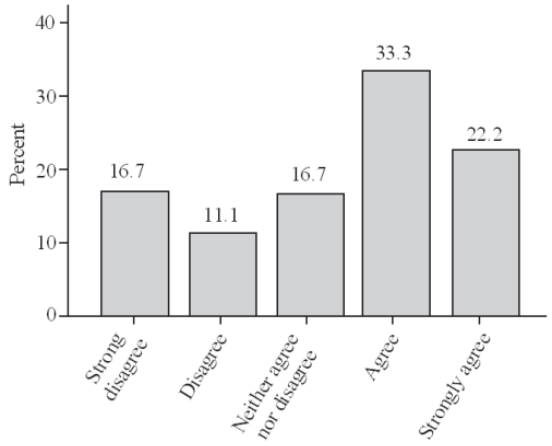

Fig. 7: I have gained the teamwork skills during my studies

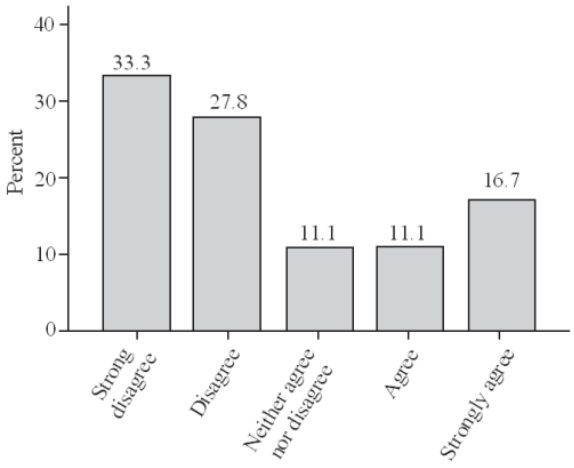

Fig. 8: I have gained the risk management skills during my studies

The frequency distribution (Fig. 3 and 4) also suggests that those students who are working in the

industry didn't find the CS290 course useful in getting promotion.

Student's response to the question of project management skill seems to be satisfactory. In Fig. 5 about 45% of the students either agree or strongly agree that they have gained the project management skills. 22% of the students were neutral and 33% are of the opinion that they didn't get the project management skills. From the frequency distribution we can say that most of the students agree that they have gained the project management skills by taking the CS290 course.

The frequency distribution in Fig. 6 clearly indicates that more students are not satisfied with thequality assurance skills gained from the CS290. The students may have not been taught the quality assurance skills in the CS290 course or may be the quality assurance skills topic was not covered in detail which indicates a problem in the course contents or may be teaching style.

Response to the question of team work skills looks satisfactory. For the most part (55%) the students agree or strongly agree that they have gained the teamwork skills. Only 27% of the students think that they didn't gain the teamwork skills (Fig. 7).

The graph in Fig. 8 shows overall dissatisfaction of the students on the risk management skills from the CS290 course. Most students (65%) are thinking that they didn't get the risk management skills from the SCS290 course as compared to those who are satisfied (27%).

In response to the question of design skills students are generally not satisfied (Fig. 9). About 44% students think they didn't get the design skills from the CS290 course and only 32% think the other way. Although the difference between the two opinions is not very significant but for best case we can say that overall students are not satisfied when they were asked if they have gained the design skills from the CS290 course.

The student's response on the requirements and documentation skills is equally distributed. Results suggests that the students are not satisfied with the requirements and documentation skills because major portion (33%) neither agree nor disagree with the question that means that they have no idea if they gained these skills from the CS290 course (Fig. 10).

The frequency distribution graph (Fig. 11) is pointing to a very important observation of the lacking of programming/coding skills. It suggests that 44% students don't agree that they gained coding skills and 17% are neutral. That means that overall we can say that the coding skills objective is not achieved by the course.

In Fig. 12 the student's response to the question of conflict resolution is almost equal splits i.e., 43% are satisfied while 39% are not satisfied.

The data analysis reveals that the student's perception of the relevance of the topics covered in the CS290 course to the SWEBOK material is satisfactory.

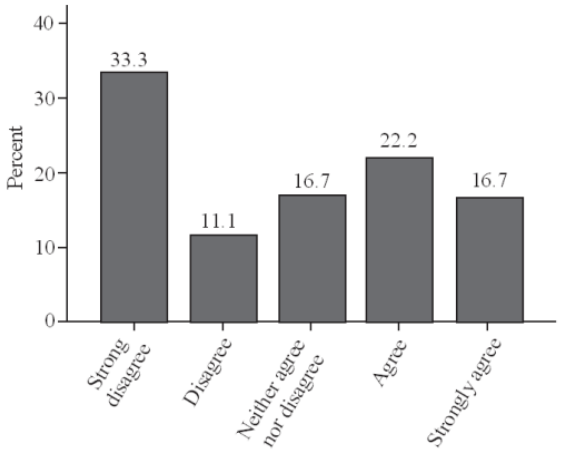

Fig. 9: I have gained the design skills during my studies; 1: Strongly disagree; 2: Disagree; 3: Neither agree nor disagree; 4: Agree; 5: Strongly agree

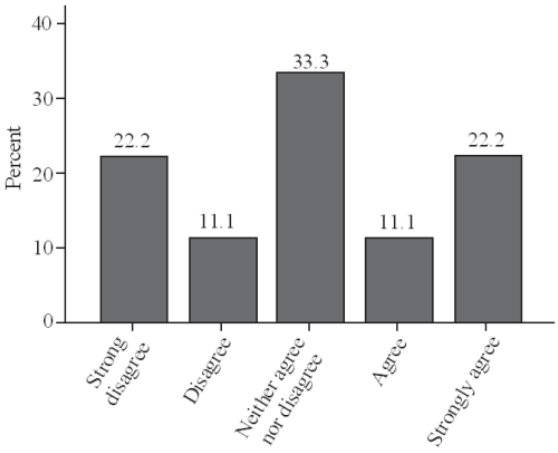

Fig. 10: I have gained requirements elicitation and documentation skills during my studies ; 1: Strongly disagree; 2: Disagree; 3: Neither agree nor disagree; 4: Agree; 5: Strongly agree

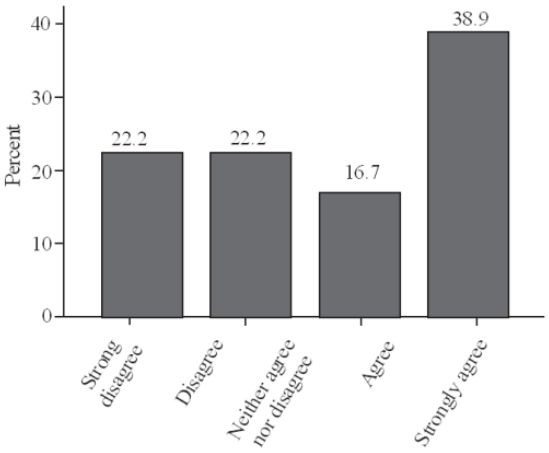

Fig. 11: I have gained the coding skills during my studies

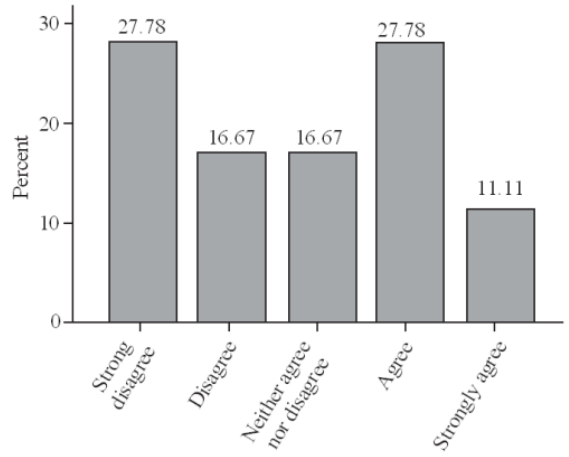

Fig. 12: I have gained the conflict resolution skills during my study

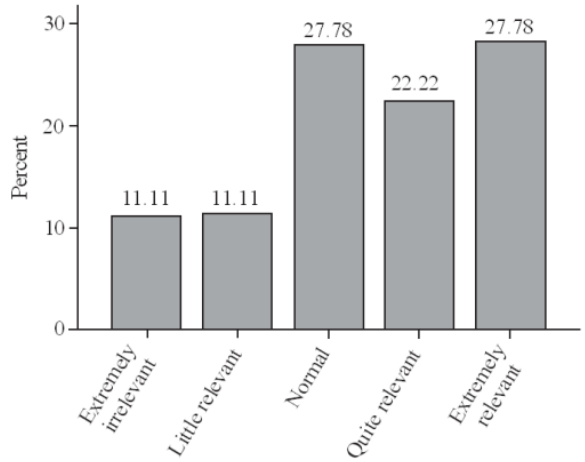

Fig. 13: Show mastery of the software engineering knowledge and skills, and professional issues necessary to begin practice as a software engineer.

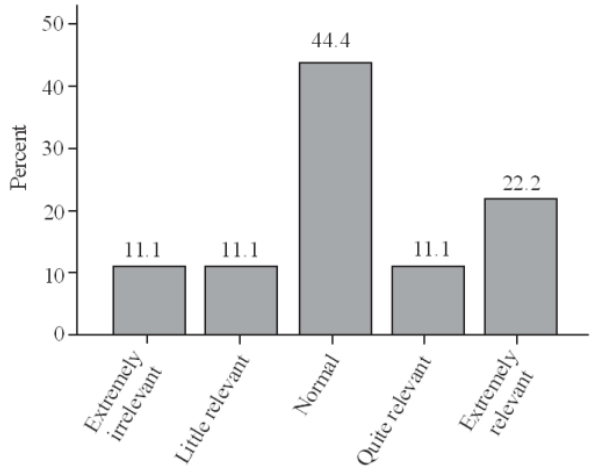

Fig. 14: Work as an individual and as part of a team to develop and deliver quality software artifacts.

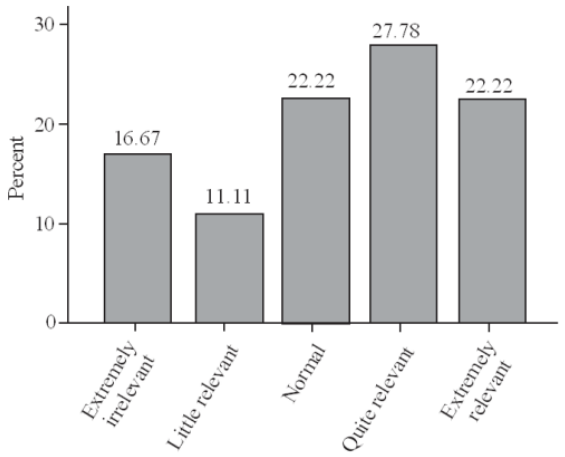

Fig. 15: Reconcile conflicting project objectives, finding acceptable compromises within limitations of cost, time, knowledge, existing systems, and organizations.

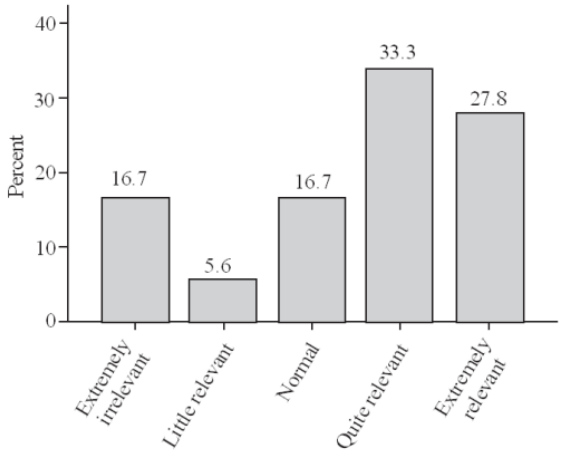

Fig. 16: Design appropriate solutions in one or more application domains using software engineering approaches that integrate ethical, social, legal, and economic concerns.

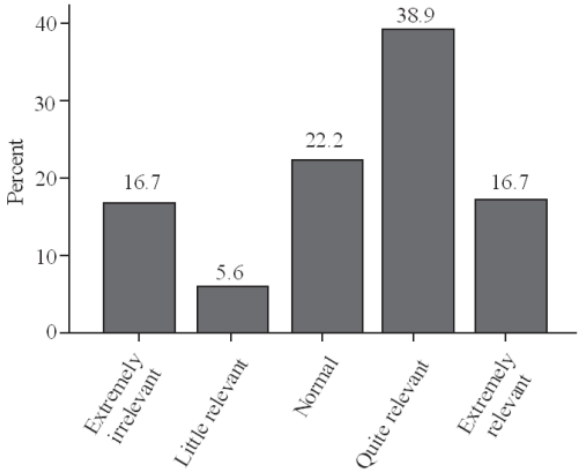

Fig. 17: Demonstrate an understanding of and apply current theories, models, and techniques that provide a basis for problem identification and analysis, software design, development, implementation, verification, and documentation.

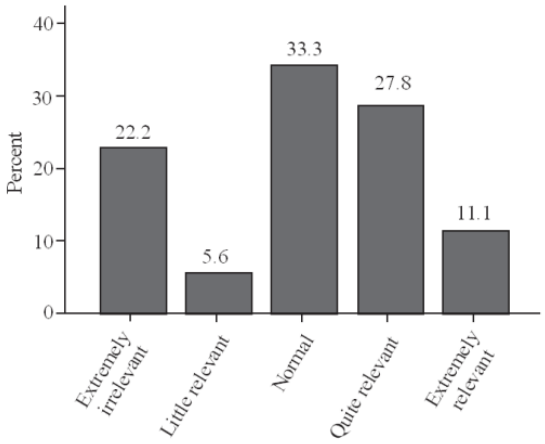

Fig. 18: Demonstrate an understanding and appreciation for the importance of negotiation, effective work habits, leadership, and good communication with stakeholders in a typical software development environment.

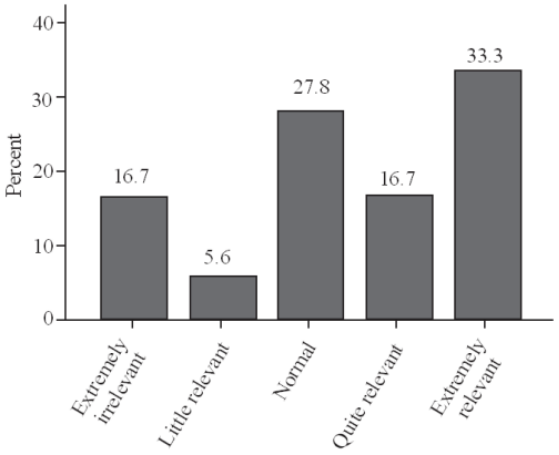

Fig. 19: Learn new model, techniques, and technologies as they emerge and appreciate the necessity of such continuing professional development.

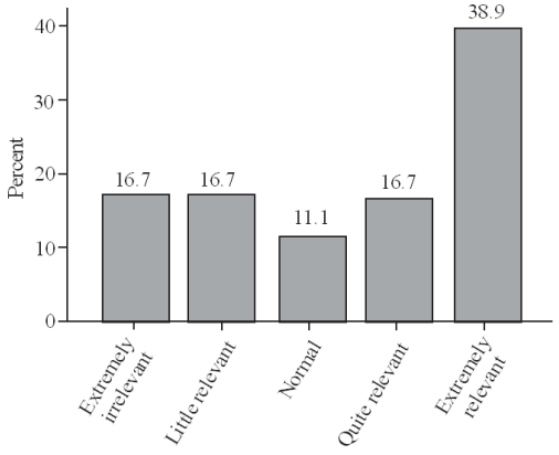

Fig. 20: Knowledge of computing essential's relevancy to industry

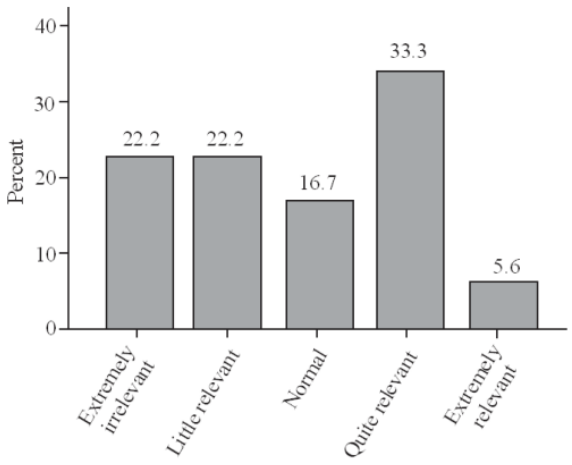

Fig. 21: Knowledge of Mathematical and Engineering Fundamental's relevancy to industry practice

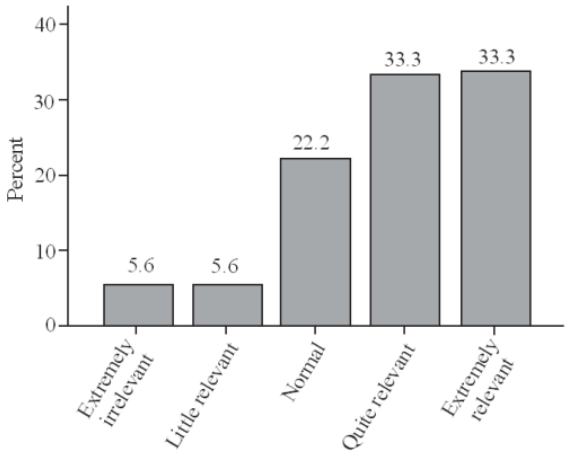

Fig. 22: Knowledge of Software Evolution's relevancy to industry

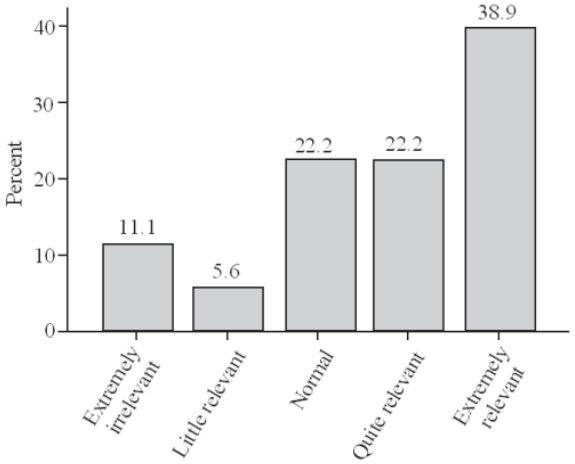

Fig. 23: Knowledge of professional practice's relevancy to industry

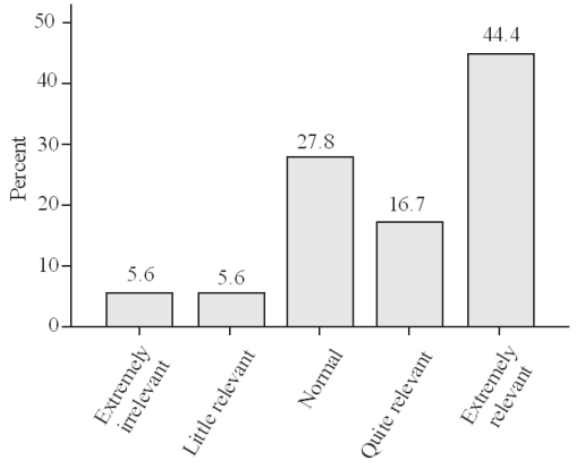

Fig. 24: Knowledge of Software Process's relevancy to industry

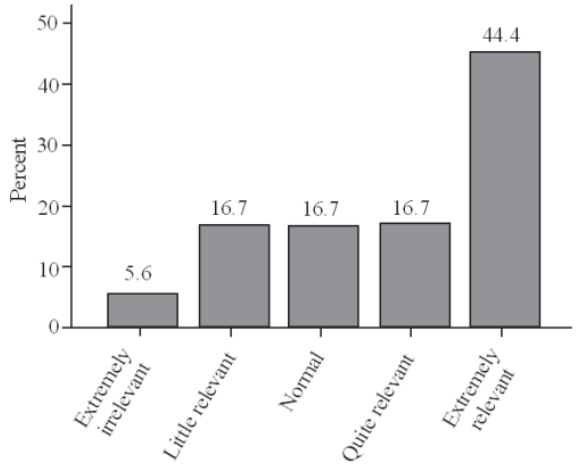

Fig. 25: Knowledge of Software Modeling and Analysis's relevancy to industry

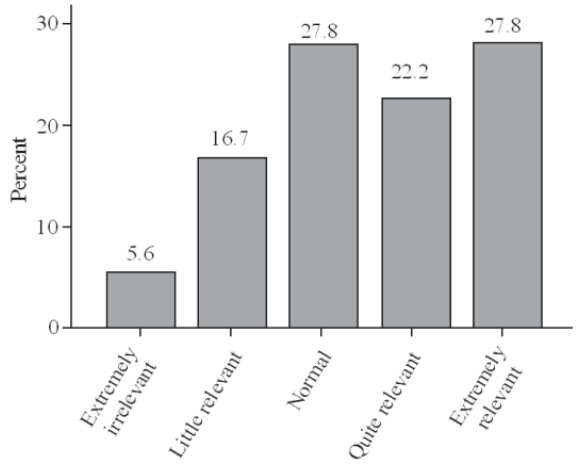

Fig. 26: Knowledge of Software Quality's relevancy to industry

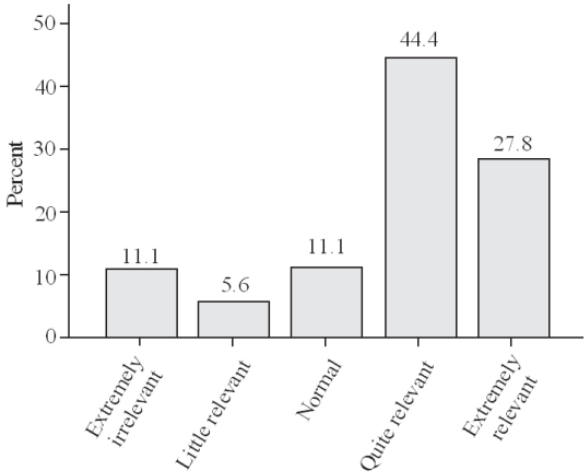

Fig. 27: Knowledge of Software Design's relevancy to industry

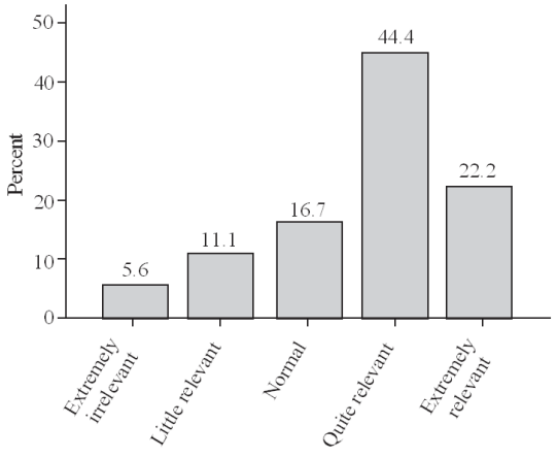

Fig. 28: Knowledge of Software Management's relevancy to industry

Most of the students are of the opinion that whatever they were taught in the CS290 course at Imam Saud University were related to the software engineering practices in the industry. The frequency distribution in Fig. 13 to 28 suggests that the topics covered in the CS290 course were relevant to the SWEBOK.

From the students response to the two sections of the survey (knowledge of the topics covered and relevancy of the topics covered to SWEBOK) it is concluded that while the topics covered are related to the SWEBOK but they (students) didn't get enough knowledge and skills to be ready for industry job.

Teaching method: After a detailed review by consulting faculty members who are teaching at the department of computer science, Imam Muhammed Ibn Saud University for the last 5 years and the senior students it was found that the teaching method of

formal authority is followed. The Formal authority is an instructor-centered approach where the instructor provides and controls the flow of content for the course. This method is the old traditional style of teaching. In today's world with new learning technologies and the net I would recommend adopting the students centered approach called the Delegator where the instructor delegates and places the control and the responsibility for learning on the students and/or groups of students.
Results of the relevance of the CS20 course.

CONCLUSION

The purpose of this study is to assess and analyze three factors. One to assess the contents of a softwareengineering (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. In brief it is recommended to review the CS290 course in light of these findings in order to make at par with the standard used world-wide (SWEBOK).

The second factor that is analyzed in this study is the perceptions (about CS290 course) of those ex-students of the Imam Muhammed Ibn Saud University who are now working in the industry. The data analysis suggests that overall the student's perception of readiness for employment by taking the software engineering course is not satisfactory. Significant numbers of students think that they are not confident to start working as a software engineer or similar role. The data indicates that the software Engineering Course (CS290) didn't fully prepare them for the job of software engineer. This result indicates that the overall perception of students is also supporting the problems (gapes) shown by in the bloom's taxonomy between the SWEBOK and the CS290 course contents (Fig. 1 to 27).

The third aspect that has been addressed in this study is the teaching method followed in the department of computer science of the Imam Muhammed Ibn Saud Islamic University. After a detailed review by consulting faculty members who are teaching at the department of computer science, Imam Muhammed Ibn Saud University for the last 5 years and the senior students it was found that the teaching method of formal authority is followed. The Formal authority is an instructor-centered approach where the instructor provides and controls the flow of content for the course. This method is the old traditional style of teaching. In today's world with new learning technologies and the net I would recommend adopting the students centered approach called the Delegator where the instructor delegates and places the control and the responsibility for learning on the students and/or groups of students.

REFERENCES

Abran, A. and J. Moore, 2009. Guide to the Software Engineering Body of Knowledge. A Stone Man Version (version.6) Retrieved form: http:// www. swebok. org/ ironrnan.

Bourque, P., R. Dupuis, A. Abran, J.W. Moore and L. Tripp, 1999. The guide to the software engineering body of knowledge. IEEE Software, 16(6): 35-44.

Felder, R.M. and R. Brent, 1996. Navigating the bumpy road to student-centered instruction. Coll. Teach., 44(2): 43-47.

Gokhale, A., 1995. Collaborative learning enhances critical thinking. J. Technol. Educ., 7(1).

Grasha, A.F., 1994. A matter of style: The teacher as expert, formal authority, personal model, facilitator, and delegator. Coll. Teach., 42(4): 142-149.

IEEE Computer Society, 2006. Guide to the SWEBOK. Retrieved form: http:// www. swcbok. org/. docurncnts/.

Johnson, R.T. and D.W. Johnson, 1994. An Overview of collaborative learning. In: Thousand, J., A. Villa and A. Nevin (Eds.), Creativity and Collaborative Learning. Brookes Publishing, Baltimore, Maryland, USA. Retrieved form: http:// www. cooperation. org/ pages/overviewpaper.html.

McPherson, K., 2005. Service Learning. New Horizons for Learning. Retrieved form: http:// www. newhorizons. org/ strategies/ service_learning/ front_service.htm.

Meyers, C. and T.B. Jones, 1993. Promoting active learning: Strategies for the college classroom. Jossey-Bass, San Francisco.

Sommerville, I., 2007. Software Engineering. 8th Edn., Addison-Wesley, Boston.

The Tuskegee University Bulletin, 2004. Courses and Programs 2004-2006. Retrieved form: http:// www. tuskegee.edu.

United States Department of Labor, 2008. Bureau of Statistics. Occupational Outlook Handbook. Retrieved form: http:// www. bls. gov/ oco/ ocos267.htm.

Wang, S.L. and S.S.J. Lin, 2007. The effects of group composition of self-efficacy and collective efficacy on computer-supported collaborative learning. Comput. Hum. Behav., 23(5): 2256-2268.

Wing, J.M., 2006. Computational thinking. Commun. ACM, 49(3): 33-35.