Benchmarking the Software Engineering Undergraduate Program Curriculum at Jordan University of Science and Technology with the IEEE Software Engineering Body of Knowledge (SWE Knowledge Areas #6–10)

Moh'd A. Radaideh

1 Introduction

The Software Engineering Undergraduate Program (SWE-Curriculum) at Jordan University of Science and Technology (JUST) recently acquired and obtained an accreditation from the Institute of Engineering and Technology (*IET*) [6]. However, the curriculum of the said program needs further expansion to ensure its readiness for any potential ABET accreditation [5] in the future as well as its readiness for training programs, professional licensing, and certification of specialties in Software Engineering. The SWEBOK-V3.0 [3] of the IEEE Computer Society [4] introduced 15 Software Engineering Knowledge Areas (SWE-KAs). Some of them are not fairly covered or addressed in the said SWE-Curriculum. Table 1 lists these 15 SWE-KAs. Table 2 lists the Software Engineering courses (SWE-Courses) of the said SWE-Curriculum [2].

This paper is a continuation of a previous paper (***P#1***) by the author in which the SWE-KAs#1–5 were addressed [1]. As shown in Table 1, this paper represents the second part (***P#2***) of the three parts of this research. It covers the second five (SWE-KAs#6–10) of the fifteen SWE-KAs. The third paper (***P#3***) shall cover the last five (SWE-KAs#11–15) of the fifteen SWE-KAs.

The SWE-KAs that are addressed in this paper are:

This paper is the second of three parts of the benchmarking research that the author has been carrying on.

Md. A. Radaideh (✉)
Jordan University of Science and Technology, Irbid, Jordan
e-mail: maradaideh@just.edu.jo

H. R. Arabnia et al. (eds.), *Advances in Software Engineering, Education, and e-Learning*, Transactions on Computational Science and Computational Intelligence, https://doi.org/10.1007/978-3-030-70873-3_7

Table 1 SWE-KAS (Software Engineering Body of Knowledge, SWEBOK-V3.0)

P#1		P#2		P#3	
SWE-KA#	SWE KAs (SWEBOK-V3.0) [3]	SWE-KA#	SWE KAs (SWEBOK-V3.0) [3]	SWE-KA#	SWE KAs (SWEBOK-V3.0) [3]
1	Software requirements	6	Software configuration management	11	SWE professional practice
2	Software design	7	SWE management	12	SWE economics
3	Software construction	8	SWE process	13	Computing foundation
4	Software testing	9	SWE models and methods	14	SWE math. Foundation
5	Software maintenance	10	Software quality	15	Engineering foundation

Table 2 The SWE courses of the SWE-Curriculum at JUST

SWE Courses at JUST (SWE-Curriculum) [1]					
SE210	Java Programming [47]	SE321	Software Requirements Eng [47]	SE430	Software Testing [50]
SE220	Software Modelling [48]	SE323	Software Documentation [48]	SE431	Software Security [51]
SE230	Fund. of Software Engineering II [43]	SE324	Software Architecture & Design [49]	SE432	Software Engineering for Web Applications [52]
SE310	Visual Programming [44]	SE326	Software engineering lab 1 [55]	SE440	Project Management [53]
SE320	Systems Analysis and Design [45]	SE471	Client/Server Programming [56]	CS318	Human-computer interaction (*Elective*) [54]
SE441	Software Quality Assurance [46]				

1. *SWE-KA#6: Software Configuration Management*. Chap. 6 of the SWEBOK-V3.0 elaborates on this SWE-KA. However, readers can refer to the many references that are listed at the end of the said chapter. Examples of these references are listed as well at the end of this paper [10–12].
2. *SWE-KA#7: Software Engineering Management*. Chap. 7 of the SWEBOK-V3.0 elaborates on this SWE-KA. However, readers can refer to the many references that are listed at the end of the said chapter. Examples of these references are listed as well at the end of this paper [13–19].
3. *SWE-KA#8: Software Engineering Process*. Chap. 8 of the SWEBOK-V3.0 elaborates on this SWE-KA. However, readers can refer to the many references that are listed at the end of the said chapter. Examples of these references are listed as well at the end of this paper [20–24].

4. *SWE-KA#9: Software Engineering Models and Methods*. Chap. 9 of the SWEBOK-V3.0 elaborates on this SWE-KA. However, readers can refer to the many references that are listed at the end of the said chapter. Examples of these references are listed as well at the end of this paper [24, 25].
5. *SWE-KA#10: Software Quality*. Chap. 10 of the SWEBOK-V3.0 elaborates on this SWE-KA. However, readers can refer to the many references that are listed at the end of the said chapter. Examples of these references are listed as well at the end of this paper [26–36].

Section 3 provides the details of our research approach (in the three parts of this research, *P#1*, *P#2*, and *P#3*) involving the reflection of the various topics of these SWE-KAs onto the various courses of the said SWE-Curriculum. It is worth mentioning that although this paper measures the coverage of the said SWE-Curriculum with the SWE-KAs, our innovative approach is general and can be applied to other Software Engineering academic programs.

The findings of this paper (*P#2*) demonstrated a decent degree of compliance in the cases of the *Software Engineering Management SWE-KA* [13–19] and the *Software Quality SWE-KA* [26–36] and a partial compliance in the cases of the *Software Configuration Management SWE-KA* [10–12], the *Software Engineering Process* SWE-KA [20–24], and the *Software Engineering Models and Methods SWE-KA* [24–25]. As the *Software Quality Assurance* is currently an elective course in the said SWE-Curriculum, this paper recommended to make this course a core rather than an elective course in the said SWE-Curriculum.

This paper is organized in several sections. Section 1 is the Introduction. Section 2 discusses the related work. Section 3 elaborates on the research methodology followed to carry on this research work. Section 4 composes five subsections where each of them elaborates on the coverage of one of the second five SWE-KAs in the SWE-Curriculum courses. Section 5 summarizes the findings of this paper, and a set of recommendations is made for possible enhancements on the said SWE-Curriculum to make it more compliant with the second five of the fifteen SWE-KAs and thus to improve its readiness for any potential ABET Accreditation in the future. Section 6 presents the conclusions of this paper.

2 Related Work

Early efforts toward organizing the teaching of Software Engineering include, but not limited to, a paper by Bernhart, M. et al., "Dimensions of Software Engineering Course Design" [7], and another one by Shaw, M. "Software Engineering Education: A Roadmap" [8]. Nevertheless, it is very important that Software Engineers read through the *The Mythical Man-Month* book of Brooks FP [9].

Qiu et al. [37] illustrated the problem-based learning approach that adopts a blended learning environment, a combination of a face-to-face learning environment and an eLearning environment for teaching undergraduate software engineering

principles as well as collaborative skills. They surveyed applying the problem-based learning approach that shows a vast student-felt comfortable learning using problem-based learning, and their educational performances were also better than anticipated.

Garousi et al. [38] investigated challenges facing fresh Software Engineering graduates early in their professional careers. Their study claimed such challenges are due to misalignment of the skills gained through their undergraduate period. They discussed the consequence of Software Engineering graduates not having practice along with soft skills in general before starting their careers, the value of certain SE activities, and abilities in SE education (especially requirements engineering, design, and testing).

Bastarrica et al. [39] surveyed software engineering students regarding relative importance and challenge of different dimensions entailed in their projects. They found out that the comparable value of soft skills develops. At the same time, that of the technical challenge falls and that the surveyed students found planning of the projects and collaboration more troublesome than they anticipated. Also, they found statistically notable evidence that, for the soft skills they have measured, the perceived corresponding relevance changes throughout the course. The surveyed students tend to undervalue the difficulty involved in teamwork before starting their Capstone Course. Therefore, they assumed that their students will be more alert to this concern while handling the upcoming projects.

Barzilay et al. [40] proposed a multidimensional Software Engineering course framework that is organized through four axes: fundamentals of SE, practices and tools, productization, and technology evolution. Their proposed work support students to have a comprehensive cross paradigm and at the same time provides practical and theoretical experience. Each axis enables an examination of Software Engineering from different perspectives. They also describe their experience of teaching the course three times in the Tel Aviv University and the academic college of Tel Aviv-Yafo, Israel.

Dekhane et al. [41] showed how to fill the gap in a project that needs knowledge in two different domains by proposing the integration of these two domains and to have one interdisciplinary project. The authors evaluated their proposed work by providing software engineering students with authentic experiences involved.

Daimi et al. [42] proposed a model that faculty can incorporate in their Software Engineering courses. The innovational approaches of brainstorming, critical thinking, case methods, problem-based learning, trimming techniques, and opportunity recognition will be introduced. Each approach will be supplemented by some examples from the Software Engineering domain. These approaches and their accompanying examples aim to develop several entrepreneurial-mindset attributes.

3 Research Methodology

The research approach followed to carry out this research (in *P#1*, *P#2*, and ***P#3***) work can be outlined in the following steps:

1. Dividing the SWE-KAs into the following two groups:
   (a) *Specialization SWE-KAs* (SWE-KAs#1–10)
   (b) *Support SWE-KAs* Knowledge Areas (SWE-KAs#10–15).
2. Splitting (*due to the size of this research work*) the *Specialization SWE-KAs* (SWE-KAs#1–10) into two parts such that the first part (*P#1*) covers the first five SWE-KAs (SWE-KAs#1–5) and the second part (*P#2*) covers the second five SWE-KAs (SWE-KAs#6–10). Consequently, the third part (*P#3*) will cover the *Support SWE-KAs* (SWE-KAs#11–15).
3. Inspecting the coverage of the SWE-KAs (*in each of* ***P#1***, ***P#2***, *and* ***P#3***) in the said SWE-Curriculum.
   (a) The coverage of the *Specialization SWE-KAs* Knowledge Areas (*P#1* and *P#2*) will be inspected across the SWE Specialization courses across the said SWE-Curriculum.
   (b) The coverage of the *Support SWE-KAs* Knowledge Areas will be inspected across the University with the College required courses across the said SWE-Curriculum.
   (c) The syllabus of each course in the said SWE-Curriculum will be carefully reviewed to figure out its coverage of the various topics of the various SWE-KAs.
   (d) The latest version of the said SWE-Curriculum (*the IET Accredited SWE-Curriculum*) will be used for this research work.
4. Classifying the coverage of each SWE-KA (in *P#1*, *P#2*, and *P#3*) in the said SWE-Curriculum into one of the following levels:
   (a) *Fully Compliant* (100%). This indicates that the concerned SWE-KA is fully covered across one or more of the courses of the said SWE-Curriculum.
   (b) *Highly Compliant* (75%–<100%). This indicates that the concerned SWE-KA is highly covered across one or more of the courses of the said SWE-Curriculum.
   (c) *Partially Compliant* (50%–<75%). This indicates that the concerned SWE-KA is partially covered across one or more of the courses of the said SWE-Curriculum.
   (d) *Poorly Compliant* (<50%). This indicates that the concerned SWE-KA is poorly covered across one or more of the courses of the said SWE-Curriculum.
5. Classifying the coverage of the main topics of each SWE-KA (in *P#1*, *P#2*, and *P#3*) in the courses learning outcomes (CLOs) of the said SWE-Curriculum. The learning outcomes are obtained from the syllabi of the courses of the said SWE-

Curriculum, which can be accessed at [1]. The CLO coverage of each SWE-KA is classified into one of the following levels:

(a) *Fully Compliant* (100%). This indicates that all main topics of the concerned SWE-KA are fully declared as course learning outcomes (CLOs) across one or more of the courses of the said SWE-Curriculum.
(b) *Highly Compliant* (75%–<100%). This indicates that most of the main topics of the concerned SWE-KA are declared as course learning outcomes (CLOs) across one or more of the courses of the said SWE-Curriculum.
(c) *Partially Compliant* (50%–<75%). This indicates that part of the main topics of the concerned SWE-KA are declared as course learning outcomes (CLOs) across one or more of the courses of the said SWE-Curriculum.
(d) *Poorly Compliant* (<50%). This indicates that few of the main topics of the concerned SWE-KA are declared as course learning outcomes (CLOs) across one or more of the courses of the said SWE-Curriculum.

6. Shortage identification and making recommendations. At the end of each part (*P#1*, *P#2*, and *P#3*), the coverage compliances (or shortages) will be identified, and recommendations will be made such that new courses shall be introduced into the curriculum or existing ones shall be enhanced and/or revised in the said SWE-Curriculum.
7. Verifying the achievement of the research prime objective. The overall purpose of this work is to facilitate the potential ABET Accreditation of the SWE Undergraduate Program of JUST.

4 SWE-KAS Coverage in the SWE-Curriculum at JUST

The following set of tables (Tables 3, 4, 5, 6, and 7) illustrate the coverage of each of the second five SWE-KAs in the various Software Engineering Courses (SWE Courses) at JUST.

4.1 Coverage of the SWE-KA#6 (Software Configuration Management)

Table 3 concludes the following:

1. The SWE-KA#6 (Software Configuration Management) seems to be *Partially Compliant* in the said SWE-Curriculum through the SE441 Software Quality Assurance course. The SE441 course addresses the various Software Configuration Management topics, but without going into details.

Table 3 SWE-KA#6 (SW Configuration Management) and its coverage in JUST SWE-Curriculum

Software Configuration Management KA (SWEBOK-V3.0)	Partially covered in
1. Management of the SCM process	SE441
Organizational context for SCM	SE441
Constraints and guidance for the SCM process	SE441
2. Planning for SCM	SE441
SCM plan	SE441
Surveillance of software configuration management	SE441
Software configuration identification	SE441
Identifying items to be controlled	SE441
Software library	
3. Software configuration	SE441
Requesting, evaluating, and approving software changes	SE441
Implementing software changes	SE441
Deviations and waivers	SE441
4. Software configuration status accounting	SE441
Software configuration status information	SE441
Software configuration status reporting	SE441
5. Software configuration	SE441
Software functional configuration audit	SE441
Software physical configuration audit	SE441
In-process audits of a software baseline	SE441
6. Software release management and delivery	SE441
Software building	SE441
Software release management auditing	SE441
7. Software configuration management tools	SE441

2. None of the main topics of the SWE-KA#6 (Software Configuration Management) contributes to the learning outcomes of the SE441 Software Quality Assurance course.

4.2 Coverage of the SWE-KA#7 (Software Engineering Management)

Table 4 concludes the following:

1. The SWE-KA#7 (Software Engineering Management) seems to be *Fully Compliant* in the said SWE-Curriculum through the SE440 Software Engineering Management course. The SE40 course covers the various Software Engineering Management topics.

Table 4 SWE-KA#7 (SWE Management) and its coverage in JUST SWE-Curriculum

SWE Management KA (SWEBOK-V3.0)	Fully Covered in	CLOs Relevance
1. Initiation and scope definition	SE 440	CLO1-to-CLO6 of the SE440
1.1 Determination and negotiation of requirements	SE 440	
1.2 Feasibility analysis	SE 440	
1.3 Process for the review and revision of requirements	SE 440	
2. Software project planning	SE 440	
2.1 Process planning	SE 440	
2.2 Determine deliverables	SE 440	
2.3 Effort, schedule, and cost estimation	SE 440	
2.4 Resource allocation	SE 440	
2.5 Risk management	SE 440	
2.6 Quality management	SE 440	
2.7 Plan management	SE 440	
3. Software project enactment	SE 440	
3.1 Implementation of plans	SE 440	
3.2 Software acquisition and supplier contract management	SE 440	
3.3 Implementation of measurement process	SE 440	
3.4 Monitor process	SE 440	
3.5 Control process	SE 440	
3.6 Reporting	SE 440	
4. Review and evaluation	SE 440	
4.1 Determining satisfaction of requirements	SE 440	
4.2 Reviewing and evaluating performance	SE 440	
5. Closure	SE 440	
5.1 Determining closure	SE 440	
5.2 Closure activities	SE 440	
6. Software engineering measurement	SE 440	
6.1 Establish and sustain measurement commitment	SE 440	
6.2 Plan the measurement process	SE 440	
6.3 Perform the measurement process	SE 440	
6.4 Evaluate measurement	SE 440	
6.5 Software engineering management tools	SE 440	
6.6 Matrix of topics vs. reference material	SE 440	

All of the main topics of the SWE-KA#7 (Software Engineering Management) contribute to the learning outcomes of the SE440 Software Engineering Management course.

4.3 Coverage of the SWE-KA#8 (SWE Process)

Table 5 concludes the following:

Table 5 SWE-KA#8 (SWE Process) and its coverage in JUST SWE-Curriculum

SWE Process KA (SWEBOK-V3.0)	Covered in	CLOs Relevance
1. Software process definition	SE441/SE324/SE230	
Software process management	SE230	
Software process infrastructure	SE230	
2. Software life cycles	SE324/SE230	
Categories of software processes	SE230/SE440	CLO4
Software life cycle models		
Software process adaptation	SE230	
Practical considerations	SE430/SE230	
3. Software process assessment and improvement	SE230	
Software process assessment models	SE230	
Software process assessment methods	SE230/SE440	CLO4 of the SE440
Software process improvement models	SE230	
Continuous and staged software process ratings	SE230	
4. Software measurement	SE324/SE230/SE441/SE323	
Software process and Product measurement	SE230	
Quality of measurement results	SE441	
Software information models		
Software process measurement techniques	SE230	
5. Software engineering process tools	SE440	CLO4 of the SE440

1. The SWE-KA#8 (Software Engineering Process) seems to be *Partially Compliant* in the said SWE-Curriculum through the following courses: (i) SE230 Fundamentals of Software Engineering, (ii) SE324 Software Architecture and Design, (iii) SE323 Software Documentation, (iv) SE440 Software Project Management, and (v) SE441 Software Quality Assurance.
2. Some of the topics of the SWE-KA#8 (SWE Process) contribute to the learning outcome CLO4 of the SE440 course, while none of these topics contributes to the learning outcomes of the SE320, SE324, SE323, and SE441 courses.

4.4 Coverage of the SWE-KA#9 (SWE Models and Methods)

Table 6 concludes the following:

1- The SWE-KA#9 (SWE Models and Methods) seems to be *Partially Compliant* in the said SWE-Curriculum through the following courses: (i) SE220 Software Modelling, (ii) SE321 Software Engineering Requirements, (iii) SE324 Software Architecture and Design, (iv) SE440 Software Project Management, and (v) SE441 Software Quality Assurance.
2- All of the main topics of the SWE-KA#9 (SWE Models and Methods) contribute to the learning outcomes of the SE220 course.
3- The last two topics of the SWE-KA#9 (SWE Models and Methods) contribute to the learning outcome CLO2 of the SE440 course.

4.5 Coverage of the SWE-KA#10 (Software Quality)

Table 7 concludes the following:

1- The SWE-KA#10 (Software Quality) seems to be *Fully Compliant* in the said SWE-Curriculum through the following courses: (i) SE430 Software Testing and (ii) SE441 Software Quality Assurance.
2- All of the main topics of the SWE-KA#10 (Software Quality) contribute to the various learning outcomes of the SE441 course.

5 Discussion and Recommendations

As indicated earlier, this paper checks the compliance of the SWE-Curriculum at JUST with the second five of the fifteen SWE Knowledge Areas (SWE-KAs#6–10) that are indicated in the Software Engineering Body of Knowledge (SWEBOK-V3.0) of the IEEE Computer Society. Table 8 provides an overall view of the

coverage of the SWE-KAs#1–10 (*SWE-KAs#1–5 comes from part 1* [1]) in the said SWE-Curriculum.

As indicated in Table 8, the compliance of the said SWE-Curriculum with the first ten SWE-KAs can be either Fully Compliant, Highly Compliant, Partially Compliant, or Poorly Compliant.

The previous paper (*P#1*) [1] recommended the introduction of the following two new courses on:

(a) *Software Construction* (based on Chap. 3 of the SWEBOK-V3.0)
(b) *Software Maintenance* (based on Chap. 5 of the SWEBOK-V3.0)

Consequently, the Author would recommend the following in this paper:

1. The settlement of the *Software Quality Assurance* course as a core course rather than an elective one in the said SWE-Curriculum.
2. The introduction of the following three new courses on:
   (a) *Software Configuration Management*: This course is strongly recommended to be based on Chap. 6 of the SWEBOK-V3.0 such that all required Software Construction related topics are covered in it.
   (b) *Software Engineering Process*: This course is strongly recommended to be based on Chap. 8 of the SWEBOK-V3.0 such that all required Software Construction related topics are covered in it.

Table 6 SWE-KA#9 (SWE Models and Methods) and its coverage in JUST SWE-Curriculum

SWE Models and Methods KA (SWEBOK-V3.0)	Covered in	CLOs Relevance
1. Modelling	SE220	CLOs of the SE220
1.1 Modelling principles	SE220	
1.2 Properties and expression of models	SE220	
1.3 Syntax, semantics, and pragmatics		
1.4 Pre-conditions, post-conditions, and invariants		
2. Types of models		
2.1 Information modelling	SE220	
2.2 Behavioral modelling	SE220	
2.3 Structure modelling	SE220	
3. Analysis of models	SE220	
3.1 Analyzing for completeness	SE441	
3.2 Analyzing for consistency	SE324	
3.3 Analyzing for correctness	SE441	
3.4 Traceability	SE321	
3.5 Interaction analysis	SE440	
4. Software engineering methods		
4.1 Heuristic methods		
4.2 Formal methods		
4.3 Prototyping methods	SE321/SE440	CLO2 of the SE440
4.4 Agile methods	SE440	

Table 7 SWE-KA#10 (SW Quality) and its coverage in JUST SWE-Curriculum

Software Quality KA (SWEBOK-V3.0)	Covered in	Declared in the following CLOs
1. Software quality fundamentals	SE441	CLO1-to-CLO10 of the SE441
1.1 Software engineering culture and ethics	SE441	
1.2 Value and costs of quality	SE441	
1.3 Models and quality characteristics	SE441	
1.4 Software quality improvement	SE441	
1.5 Software safety	SE441	
2. Software quality management processes	SE441	
2.1 Software quality assurance	SE441	
2.2 Verification & Validation	SE441	
2.3 Reviews and audits	SE441	
3. Practical considerations	SE441/SE430	
3.1 Software quality requirements	SE441	
3.2 Defect characterization	SE441	
3.3 Software quality management techniques	SE441	
3.4 Software quality measurement	SE441	
4. Software quality tools	SE441	

(c) *Software Models and Methods*: This course is strongly recommended to be based on Chap. 9 of the SWEBOK-V3.0 such that all required Software Construction related topics are covered in it.

6 Conclusions

This paper reflects the compliance of the SWE-Curriculum at JUST with the SWE-KAs#6–10 of the Software Engineering Body of Knowledge of the IEEE Computer Society.

The SWE-KA#7 Software Engineering Management and the SWE-KA#10 Software Quality are fully covered (*Fully Compliant*) in the said SWE-Curriculum. While the remaining three SWE Knowledge Areas (SWE-KA#6 Software Configuration Management, SWE-KA#8 Software Engineering Process, and SWE-KA#9 Software Engineering Models and Methods) are partially covered (*Partially Compliant*) in the said SWE-Curriculum. Therefore, the author recommended the introduction of three new courses on (i) *Software Configuration Management*, (ii) *Software Engineering Process*, and (iii) *Software Engineering Models and Methods* into the said SWE-Curriculum. These three new courses come in addition to the two courses that were recommended in the previous part of this research (*P#1*) [1] (e.g., *Software Construction* and *Software Maintenance* courses). Also, in this paper, the author recommended that the existing *Software Quality Assurance* course becomes a core course rather than an elective one in the said SWE-Curriculum.

Table 8 The overall coverage of the SWE-KAS#1–10 in JUST SWE-Curriculum

		Coverage in the SWE-Curriculum at JUST															
	SWE-KA	*Coverage*	SE210	SE220	SE225	SE230	SE310	SE320	SE321	SE324	SE323	SE430	SE431	SE432	SE440	SE441	SE471
Part#1 [1]	Software requirements	*Fully compliant*				√			√			√					
	Software design	*Highly compliant*	√	√		√	√			√				√			
	Software construction	*Partially compliant*	√		√	√	√		√	√		√			√	√	√
	Software testing	*Fully compliant*				√						√				√	
	Software maintenance	*Partially compliant*				√				√						√	
Part#2 [this paper]	Software configuration	*Partially compliant*														√	
	SWE management	*Fully compliant*													√		
	SWE process	*Partially compliant*				√				√	√				√	√	
	SWE models and methods	*Partially compliant*		√					√	√					√	√	
	Software quality	*Fully compliant*										√				√	

Acknowledgments The author would like to thank Dr. Ahmed Shatnawi for his valuable input and careful proofreading of the final version of this paper.

References

1. M. Radaideh, Benchmarking the Software Engineering Undergraduate Program Curriculum at Jordan University of Science and Technology with the IEEE Software Engineering Body of Knowledge (Software Engineering Knowledge Areas #1–5); Accepted in the "*The 18th International Conference on Software Engineering Research and Practice (SERP'20)*", and in the Research Book Series "*Transactions on Computational Science & Computational Intelligence*" https://www.springer.com/series/11769
2. The curriculum of the software engineering undergraduate program at Jordan University of Science and Technology; http://www.just.edu.jo/FacultiesandDepartments/it/Departments/SE/SiteAssets/Pages/Programs/IET-SE_English_StudyPlan2016.pdf
3. P. Bourque, R. Dupuis, *Guide to the Software Engineering Body of Knowledge* (IEEE Computer Society, Los Alamitos, 2004) SWEBOK-V3.0; https://www.computer.org/education/bodies-of-knowledge/software-engineering
4. IEEE Computer Society; https://www.computer.org/
5. ABET Accreditation; https://www.abet.org/accreditation/
6. Institute of Engineering and Technology Site – IET Accreditation; https://www.theiet.org/career/accreditation/academic-accreditation/
7. M. Bernhart, T. Grechenig, J. Hetzl, & W. Zuser, Dimensions of software engineering course design, ICSE 2006, Shanghai, China, May 20–28, pp. 667–672 (2006)
8. M. Shaw, Software engineering education: A roadmap. ICSE – Future SE Track, 371–380 (2000)
9. F.P. Brooks, *The Mythical Man-Month, Anniversary Edition* (Addison-Wesley, Boston, 1975)
10. IEEE Std. 828-2012, *Standard for Configuration Management in Systems and Software Engineering* (IEEE, 2012)
11. A.M.J. Hass, *Configuration Management Principles and Practices*, 1st edn. (Addison-Wesley, 2003)
12. CMMI Product Team, *CMMI for Development, Version 1.3* (Software Engineering Institute, 2010).; http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=9661
13. Project Management Institute, *A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide)*, 5th edn. (Project Management Institute, 2013)
14. Project Management Institute and IEEE Computer Society, *Software Extension to the PMBOK® Guide*, 5th edn. (Project Management Institute, 2013)
15. R.E. Fairley, *Managing and Leading Software Projects* (Wiley-IEEE Computer Society Press, 2009)
16. B. Boehm, R. Turner, *Balancing Agility and Discipline: A Guide for the Perplexed* (Addison-Wesley, 2003)
17. IEEE Std. 15939-2008, *Standard Adoption of ISO/IEC 15939:2007 Systems and Software Engineering—Measurement Process* (IEEE, 2008)
18. J. McGarry et al., *Practical Software Measurement: Objective Information for Decision Makers* (Addison-Wesley Professional, 2001)
19. J. McDonald, *Managing the Development of Software Intensive Systems* (John Wiley and Sons, Inc., 2010)
20. R.E. Fairley, *Managing and Leading Software Projects* (Wiley-IEEE Computer Society Press, 2009)
21. J.W. Moore, *The Road Map to Software Engineering: A Standards-Based Guide* (Wiley-IEEE Computer Society Press, 2006)

22. Project Management Institute and IEEE Computer Society, *Software Extension to the PMBOK® Guide*, 5th edn. (Project Management Institute, 2013)
23. D. Gibson, D. Goldenson, K. Kost, *Performance Results of CMMI-Based Process Improvement* (Software Engineering Institute, 2006) http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=8065
24. ISO/IEC 15504-1:2004, *Information Technology—Process Assessment—Part 1: Concepts and Vocabulary* (ISO/IEC, 2004)
25. J.M. Wing, A specifier's introduction to formal methods. Computer **23**(9), 8 (1990)., 10–23
26. P.B. Crosby, *Quality Is Free* (McGraw-Hill, 1979)
27. W. Humphrey, *Managing the Software Process* (Addison-Wesley, 1989)
28. S.H. Kan, *Metrics and Models in Software Quality Engineering*, 2nd edn. (Addison-Wesley, 2002)
29. ISO/IEC 25010:2011, *Systems and Software Engineering—Systems and Software Quality Requirements and Evaluation (SQuaRE)—Systems and Software Quality Models* (ISO/IEC, 2011)
30. IEEE, *P730™/D8 Draft Standard for Software Quality Assurance Processes* (IEEE, 2012)
31. F. Bott et al., *Professional Issues in Software Engineering*, 3rd edn. (Taylor & Francis, 2000)
32. D. Galin, *Software Quality Assurance: From Theory to Implementation* (Pearson Education Limited, 2004)
33. ISO 9000:2005, *Quality Management Systems—Fundamentals and Vocabulary* (ISO, 2005)
34. IEEE Std. 1012-2012, *Standard for System and Software Verification and Validation* (IEEE, 2012)
35. IEEE Std. 1028-2008, *Software Reviews and Audits* (IEEE, 2008)
36. K. Wiegers, *Peer Reviews in Software: A Practical Guide* (Addison-Wesley Professional, 2001)
37. M. Qiu, L. Chen, A problem-based learning approach to teaching an advanced software engineering course. In 2010 Second International Workshop on Education Technology and Computer Science, 2010
38. V. Garousi, G. Giray, E. Tuzun, C. Catal, M. Felderer, Closing the gap between software engineering education and industrial needs. IEEE Softw. (2019)
39. M.C. Bastarrica, D. Perovich, M.M. Samary, What can students get from a software engineering capstone course? in *2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering Education and Training Track (ICSE-SEET)*, (IEEE, 2017), pp. 137–145
40. O. Barzilay, O. Hazzan, A. Yehudai, A multidimensional software engineering course. IEEE Trans. Educ. **52**(3), 413–424 (2009)
41. S. Dekhane, M.Y. Tsoi. Work in progress—Inter-disciplinary collaboration for a meaningful experience in a software development course. In *2010 IEEE Frontiers in Education Conference (FIE)* (IEEE, 2010), pp. S1D–1
42. K. Daimi, Strengthening elements of teamwork, innovation, and creativity in a software engineering program. J. Eng. Entrep. **3**(1), 35–50 (2012)
43. SE230 Fundamentals of Software Engineering; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE230.pdf
44. SE310 Visual Programming; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE310.pdf
45. SE320 System Analysis and Design; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE320.pdf
46. SE321 Software Requirements Engineering; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE321.pdf
47. SE323 Software Documentation; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE323.pdf
48. SE324 Software Architecture & Design; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE324.pdf
49. SE430 Software Testing; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE430.pdf
50. SE431 Software Security; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE431.pdf

51. SE432 Software Engineering for Web Applications; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE432.pdf
52. SE440 Project Management; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE440.pdf
53. CS318 Human-Computer Interaction; http://www.just.edu.jo/~ahmedshatnawi/syllabus/CS318.pdf
54. SE471: Client Server Programming; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE471.pdf
55. SE326: Software Engineering Lab; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE326.pdf
56. SE441: Software Quality Assurance; http://www.just.edu.jo/~ahmedshatnawi/syllabus/SE441.pdf

Moh'd A. Radaideh is currently an *Associate Professor* with the Department of Software Engineering, Jordan University of Science and Technology. He is a Senior Member of the IEEE, the IEEE Computer Society, and the IEEE Education Society. He received his BENG & MENG Degrees in Electrical and Computer Engineering from Yarmouk University and Jordan University of Science and Technology, consequently in 1987 and 1989. He obtained his Ph.D. degree in Electrical and Computer Engineering (Software Engineering) from McMaster University (Canada) in 2000.

Profile: http://www.just.edu.jo/admissionuploads/staff_cv/maradaideh.pdf.