What Are Massive Open Online Courses (MOOCs) Teaching about Software Testing?

Džiugaite, Neda ; Ardic, Baris; Zaidman, Andy

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

What Are Massive Open Online Courses (MOOCs) Teaching About Software Testing?

Neda Džiugaitė
neda.dziugaite@gmail.com
Delft University of Technology
The Netherlands

Baris Ardic
b.ardic@tudelft.nl
Delft University of Technology
The Netherlands

Andy Zaidman
a.e.zaidman@tudelft.nl
Delft University of Technology
The Netherlands

ABSTRACT

Software testing is a necessary aspect of software development. With high expectations placed on software testers and a shortage of qualified professionals, Massive Open Online Courses (MOOCs) have emerged as a potential solution to improve software testing education. MOOCs provide accessible education, bridging the gap between formal education and industry expectations. We investigate key aspects of and compare concepts of software testing MOOCs with university curricula and industry expectations. The findings show that a MOOC on average covers more concepts than a single university course. Additionally, MOOCs align well with what the industry expects from software testing practitioners.

KEYWORDS

software testing, education, MOOCs

1 INTRODUCTION

Software testing is necessary to assure the dependability and quality of software programs [16]; it encompasses many different practices, tools, and techniques [21], and Florea and Stray found that software testers are not expected to specialize in one particular activity, such as performance, automation, or test management, but rather to be proficient in a wide range of testing-related skills [13]. This puts high expectations on practitioners and reinforces the lack of qualified professionals in the software testing field [24]. Additionally, one of the problems in software testing education is that formal education does not align with the industry demand [24]. Massive Open Online Courses (MOOCs) can be one of the solutions to bridge the gap between formal education and industry expectations. MOOCs have been a popular medium to learn new skills or enhance one's existing skillset [22], providing access to high-quality education at a fraction of the cost of formal education [7]. MOOCs can offer an organized and thorough review of software

testing principles and procedures, enabling practitioners to fill any knowledge gaps they may have to meet industry expectations.

This paper aims to provide insight into how MOOCs contribute to software testing knowledge and education. To gain insights into their role in this field, we conducted a study of software testing courses offered through MOOCs. The results of the study were then used to compare concepts discussed in software testing MOOCs to university curricula and industry expectations.

RQ1 What are the key aspects of software testing MOOCs?
    RQ1.1 What information is provided for the user before starting a MOOC?
    RQ1.2 What are the entry requirements for software testing MOOCs?
    RQ1.3 What teaching techniques are most common in software testing MOOCs?
    RQ1.4 What are the most common software testing concepts discussed in MOOCs?
RQ2 Do the concepts taught in MOOCs align with what is being taught in universities?
RQ3 Do the concepts taught in MOOCs align with what the industry expects from software testing practitioners?

2 RELATED WORK

University curricula. Ardic and Zaidman analyzed university curricula to identify key software testing concepts, identifying *test process* and *test type* as the most frequent aspects [2].

Industry expectations. Florea and Stray assessed job advertisements to determine industry-required software testing skills, noting a high demand for specific skills [13]. Cerioli et al. reviewed 5 million job ads, discovering the industry's preference for automated testing, highlighting the value of proficiency in tools like Selenium [8]. Kassab et al. studied 1000 job postings in the US, revealing a high demand for regression and automated testing skills [17].

MOOCs. Sharov et al. evaluated online Python courses on MOOC platforms, determining their effectiveness for learning Python [23]. Unlike Sharov et al., who focused on quantitative aspects like course availability and pricing, this study delves deeper into course contents. Bali assessed pedagogical practices in MOOCs and their impact on learner engagement, highlighting that engagement is more beneficial than MOOC structure or instructor tasks [4]. Similarly, Koedinger et al. found that interactive activities in MOOCs enhance learning [18]. Ma et al. reviewed the literature to identify factors influencing online course purchases, linking course, instructor, and platform attributes to consumer decisions [19].

Provider	After search and filtering	Dedicated testing MOOCs
Coursera	72	3+5*
EdX	20	3
FutureLearn	20	3
Udemy	3141	3+5*
Saylor	7	0
Khan Academy	7	0
Udacity	7	1

* - There were more dedicated software testing MOOCs, therefore heuristics were used to select the courses.

Table 1: Course selection from each of the 7 providers.

3 METHODOLOGY

This section details the rationale for choosing providers and courses and describes the data collection methods.

3.1 Course selection

The providers were selected based on the list of free online course platforms provided by the European Job Mobility Portal (EURES) [3] in addition to Udacity, because of its popularity [12, 14, 20]. The selected providers are listed in Table 1.

EdX, FutureLearn, Udacity, Saylor, and Khan Academy yielded a manageable number of courses after searching for "software testing," with only EdX (3 courses), FutureLearn (3 courses), and Udacity (1 course) offering relevant content, while Saylor and Khan Academy had none. Coursera, with a larger catalogue, required a manual review of 72 MOOCs using search and filters, leading to the selection of the top 3 most highly-rated courses. Courses lacking reviews or with hidden ratings were excluded, as were short projects and guided projects due to their limited scope. Udemy presented 10,000 potential courses. Filtering by a rating of 4.5 or above and sorting by the number of reviews narrowed this to the first 3 relevant courses.

To ensure a balanced analysis, no more than 3 courses per provider were selected, focusing on course descriptions, entry requirements, and teaching techniques, which could be influenced by the platform's features and presentation style.

For a broader analysis of topics, which are less platform-dependent, an additional 10 courses were included: 5 from Coursera, chosen without considering reviews to include courses without ratings, and 5 from Udemy, selected by the previously mentioned criteria.

3.2 Data collection

To identify prevalent concepts in software testing MOOCs, we evaluated the selected courses, noting metadata like price, duration, entry-level, and prerequisites. Entry levels were deduced from prerequisites or descriptions if unspecified. We also recorded ratings, the number of reviews, and enrolment figures when available.

Information accessible pre-enrolment was catalogued: teaching methods — including videos, reading materials, quizzes, practical tasks, and discussion prompts — were documented, although for 4 courses was unavailable due to access restrictions.

For topic analysis, 23 courses were examined. Keywords on topics and tools came from syllabi, course content, learning objectives, and descriptions, typically extracted from video, section titles, and
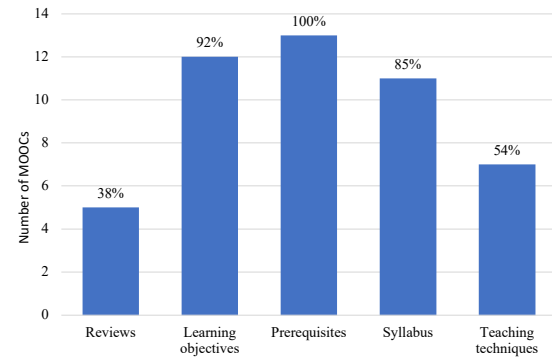

Figure 1: Results of what is presented for users in the course descriptions of software testing MOOCs before enrolling.

summaries. If these were vague or absent, keywords were drawn from module openings where instructors outlined topics.

4 RESULTS OF THE MOOC ANALYSIS

This section will discuss the results of the study that was conducted. The full dataset with all of the results can be found at [10].

4.1 Course descriptions

Figure 1 shows the information that is presented to a user before starting a course. We observe that almost all courses provide learning objectives, prerequisites, and syllabi, however, these vary in detail. Most courses have fairly vague prerequisites, some of which are inaccurate as mentioned in course reviews, or observed after analyzing the course content. Seven of the courses mention what teaching techniques are used in the course, although often incomplete. There are no mentions of discussion prompts or peer-graded assignments. Additionally, it is rarely mentioned what labs are available. Lastly, reviews are only available for 5 of the courses. This is due to the fact that only Coursera and Udemy have review functionality for the courses in this study. Even among these 2 providers, there was 1 course which did not allow the user to see the reviews, the user could only observe the overall course rating.

4.2 Entry requirements

Eight of the courses explicitly state the entry level, mentioning beginner, intermediate, and advanced levels. The level for the other 5 courses was deduced from the prerequisites listed.

Four of the 13 courses were tagged as beginner courses, 8 courses were of intermediate level, and 1 course was labeled advanced. Beginner courses have little to no prerequisites, most of them listing basic computer skills as the main prerequisite. The prerequisites for intermediate courses vary: half of these courses require or recommend the student to know a certain programming language. Other prerequisites include the ability to install and run an IDE, mathematical and logical reasoning, and shell scripting. Lastly, the advanced course is a follow-up to one of the intermediate courses, therefore the prerequisites are to be familiar with that course.

4.3 Teaching techniques

As can be seen in Figure 2 videos are the most popular teaching technique with 8 out of 9 courses using them. Reading and practical
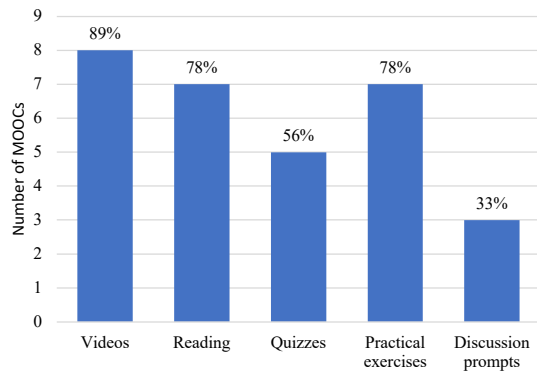
Figure 2: Teaching techniques found in testing MOOCs.

exercises are commonly employed as well. Quizzes are the less popular teaching technique with only 5 courses providing them. Additionally, a few courses make use of discussion prompts and peer-graded assignments. Discussion prompts allow students to post their thoughts on a message board in the platform and discuss with peers or instructors. Practical exercises are provided in various ways. Half of the courses that provide practical exercises include a dedicated lab environment. Only one course offered automatic grading, while, others usually provided example solutions.

4.4 Topics covered

To understand the topics that these MOOCs cover, we extracted keywords from the learning objectives, syllabus, course descriptions, and course content. We observe that the most common keywords in software testing MOOCs are automation (61%), unit testing (57%), test levels (52%) and test coverage (43%).

In order to have a better overview of the data collected, we categorized the keywords using the taxonomy of software testing skills as defined by Florea and Stray [13]. In addition to the skills listed in the taxonomy, it is necessary to clarify how we classified keywords that were not listed in the taxonomy. Keywords such as automation, test-driven development, and behaviour-driven development were included in the testing process category. Various types of testing such as mutation testing, performance testing, load testing, API testing, and so on were included in the test type category.

The results after categorization can be found in Figure 3. Almost all courses cover at least some aspect of the test process category; more than half covered test levels, test types, and various software testing tools. The least covered topic was static testing.

4.5 Comparing results

We compare our findings of the most commonly discussed software testing aspects in MOOCs to the ones in (1) dedicated software testing university courses and (2) to requirements from software testing job adverts. Ardic and Zaidman have analyzed university curricula in order to figure out what software testing topics are commonly discussed in dedicated software testing courses [2]; they categorize their results using the software skills taxonomy by Florea and Stray [13]. Furthermore, we compare results with a study by Florea and Stray in which they analyzed job advertisements in order to find what software testing skills are most in demand in the industry [13]. The comparison can be found in Figure 4.
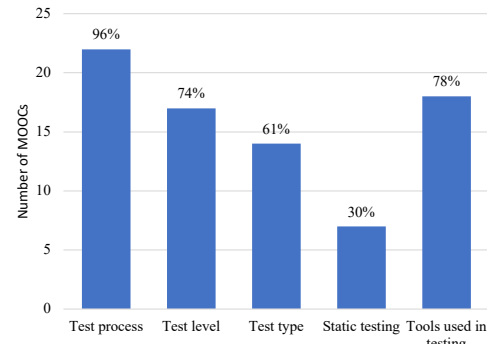

Figure 3: Concepts discussed in software testing MOOCs, categorized using the taxonomy of software testing skills[13].

The categories of test process, tools used in testing, and test level are most covered by MOOCs. The difference in coverage for test process and tools is higher for MOOCs and universities than it is for MOOC and industry needs. Test levels are much more popular in MOOCs than they are in universities or required in job advertisements. The test type category shows similar coverage of around 60% for MOOCs, university courses, and job advertisements. Static testing was the only category that had a significantly lower coverage in MOOCs compared to university courses.

5 DISCUSSION & CONCLUSION

This section revisits the research questions, examines possible threats to validity of our study, and presents the conclusion.

5.1 Revisiting the Research Questions

RQ1.1: *What information is provided for the user before starting a MOOC?* Before enrolling, the majority of courses offer key details like learning objectives, prerequisites, and syllabi, which are crucial for users in aligning the course with their learning goals and current knowledge base, as users tend to select online courses based on their expectations for learning outcomes and their need for knowledge improvement [15]. About 85% of the courses are either free, or have a free trial, which makes it easier to assess a course, as details regarding practical exercises are rarely provided from the course description. Detailed teaching techniques are disclosed in only half of the courses, potentially leaving users guessing about the nature of instructional delivery, despite the predictability of teaching techniques in MOOCs [23]. Reviews, which can significantly affect enrollment and purchase decisions as indicated by Ma et al. [19], are available for only 38% of courses. This low percentage may be attributed to the limited functionality for reviews on the MOOC platforms considered in this study.

RQ1.2: *What are the entry requirements for software testing MOOCs?* The majority of courses were intermediate level, assuming users had programming knowledge. These courses aim to bridge the gap between formal education and industry requirements, assuming students have programming experience and basic testing knowledge. A few beginner courses, requiring minimal prior knowledge, focus on high-level software testing topics like test planning and defect reporting, making them accessible without prior software engineering education. An advanced course served as a sequel
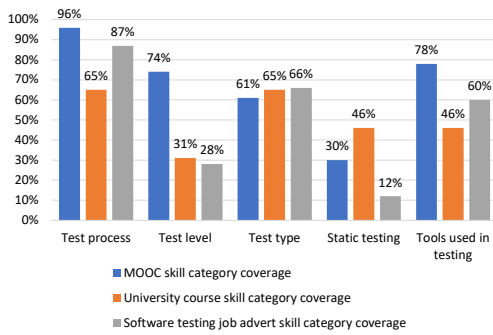
Figure 4: Concepts discussed in software testing MOOCs, compared with universities [2] and to software testing job advertisements [13], categorized using the taxonomy of software testing skills [13].

to an intermediate one on the same platform, expecting familiarity with the earlier content and practical testing experience.

RQ1.3: *What teaching techniques are most common in software testing MOOCs?* Most courses incorporate video lectures, reading materials, and practical exercises, with practical exercises offering more significant learning outcomes [18]. About half of the courses use quizzes as simple assessments, which may not reflect real-world scenarios accurately [4]. Thus, it is encouraging that many courses prioritize complex practical exercises, either exclusively or alongside quizzes. A few courses employ discussion prompts, forums, and peer-graded exercises, enabling student interaction and collaboration. These features serve as avenues for students to help each other, mitigating the absence of instructor interaction [4].

RQ1.4: *What are the most common software testing concepts discussed in MOOCs?* Keywords extracted highlight common topics in software testing courses: automation, unit testing, test levels, and test coverage. Automation is a prominent topic due to the costliness of testing [6, 11]. Other topics are fundamental to software testing; categorizing these reveals that almost all courses cover aspects of the testing process, and over half address topics like test levels, test types, and testing tools or frameworks. Static testing receives less attention, possibly due to limitations in performance [1].

RQ2: *Do the concepts taught in MOOCs align with what is being taught in universities?* Topic coverage for most categories is higher in MOOCs than in universities. This indicates that an average MOOC covers more different topics than a dedicated software testing course. However, it is unclear how in-depth these topics are covered. Software testers are required to be proficient in a wide array of different software testing skills, hence higher topic coverage is beneficial for someone preparing to go into industry [13].

RQ3: *Do the concepts taught in MOOCs align with what the industry expects from software testing practitioners?* MOOCs effectively align with the skill requirements of the software testing industry. The majority of software skills sought in job advertisements are covered in MOOCs, often with similar or higher percentages.

Automation, a highly sought-after skill in the industry [8, 13, 17], emerges as the most frequently mentioned keyword (%61) in software testing MOOCs. The most in-demand testing type, regression testing [8, 17], is covered by 26% of MOOCs. Other popular testing types in MOOCs include performance (22%), functional (17%),

and security testing (17%), mirroring their prevalence in job advertisements [8, 17]. The Selenium tool, frequently required in job advertisements [8, 13, 17], receives substantial (30%) coverage.

In conclusion, MOOC content aligns with industry demands for software testing skill categories, keywords, testing types, and tools. This aligns with practitioners seeking to enhance their qualifications through MOOCs [23].

5.2 Threats to Validity

This study acknowledges potential threats to its validity.

External Validity: The study analyzed a relatively small sample of 13 courses for course descriptions, entry requirements, and teaching techniques, as well as 23 courses for software testing topics, all selected from 7 MOOC platforms. While this sample offers insights into popular software testing MOOCs, it may not fully represent the diversity of available courses in this field.

Internal validity. The course selection process employed a systematic strategy using defined search terms, filters, and sorting options, aiming to ensure replicability [10]. The filters chosen could have inadvertently excluded some relevant courses. The study focused on popular courses and providers, mirroring the search experience of an average user looking for a software testing MOOC. Thus, the likelihood of missing pertinent MOOCs is low. Course ratings or content may change over time, impacting their inclusion or exclusion from the sample. To mitigate this effect, we added data collection dates to the replication package.

Construct Validity: Relying on MOOC descriptions and content overviews for analysis presents potential threats. These may not represent actual content, leading to the potential omission of keywords, concepts, or techniques. While the study clearly defined its data collection methods, this threat remains partially unaddressed, suggesting potential areas for further refinement. Moreover, experiences with higher education courses can also differ by region resulting in higher [5] or lower [2] levels of coverage of subjects.

5.3 Conclusion

Our findings reveal that dedicated software testing MOOCs are typically well-structured, offering essential information such as learning objectives, prerequisites, and syllabi. We observe that the majority of these courses are designed for intermediate-level learners. The presence of practical exercises, a key factor for effective learning, was a common feature in the studied courses.

MOOC content appears to align with industry requirements, as indicated by the prevalence of keywords, tools, and testing types consistent with industry demand. Conversely, MOOCs typically cover a broader spectrum of topics compared to university courses.

A next logical step for this line of work is to add qualitative insights from participants of these software testing MOOCs, to further deepen our understanding of alignment with industry needs.

ACKNOWLEDGMENTS

REFERENCES

[1] P. Anderson. 2008. The Use and Limitations of Static-Analysis Tools to Improve Software Quality. *CrossTalk-Journal of Defense Software Engineering* 21 (06 2008).

[2] Baris Ardic and Andy Zaidman. 2023. Hey Teachers, Teach Those Kids Some Software Testing. In *Proceedings of the Fifth ICSE Workshop on Software Engineering Education for the Next Generation (ICSE SEENG)*. IEEE, 9–16.

[3] European Labour Authority, Social Affairs Directorate-General for Employment, and Inclusion. 2020. Top 6 free platforms for online courses. Available: https://eures.ec.europa.eu/top-6-free-platforms-online-courses-2020-05-11_en. [Online].

[4] Maha Bali. 2014. MOOC Pedagogy: Gleaning Good Practice from Existing MOOCs. *Journal of Online Learning and Teaching* 10, 1 (2014), 44–56.

[5] Ayodele A. Barrett, Eduard Paul Enoiu, and Wasif Afzal. 2023. On the Current State of Academic Software Testing Education in Sweden. In *2023 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. 397–404. https://doi.org/10.1109/ICSTW58534.2023.00073

[6] Moritz Beller, Georgios Gousios, Annibale Panichella, Sebastian Proksch, Sven Amann, and Andy Zaidman. 2019. Developer Testing in the IDE: Patterns, Beliefs, and Behavior. *IEEE Trans. Software Eng.* 45, 3 (2019), 261–284.

[7] Inma Borrella, Sergio Caballero-Caballero, and Eva Ponce-Cueto. 2022. Taking action to reduce dropout in MOOCs: Tested interventions. *Computers & Education* 179 (2022).

[8] Maura Cerioli, Maurizio Leotta, and Filippo Ricca. 2020. What 5 Million Job Advertisements Tell Us about Testing: A Preliminary Empirical Investigation. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing (SAC)*. ACM, 1586—-1594.

[9] Neda Džiugaitė. 2023. *Mining massive open online courses (MOOCs) for software testing knowledge.* Bachelor's Thesis. http://resolver.tudelft.nl/uuid:d53f052e-4016-425e-adfe-73963c33b1d9

[10] Neda Džiugaitė. 2023. Dataset of Software Testing Massive Open Online Courses (MOOCs). (2023). [Online]. Available at: https://doi.org/10.4121/1c8546e4-ecfd-4c60-986e-42198d5e0d5b.v1.

[11] Christof Ebert. 2011. *Global Software and IT: A Guide to Distributed Development, Projects, and Outsourcing* (1st ed.). Wiley-IEEE Computer Society Pr.

[12] EduTechReviews. 2017. The 9 best MOOCs platforms. *Medium* (2017). [Online]. Available at: https://medium.com/@edutechreviews/the-9-best-moocs-platforms-631846001b6e.

[13] Raluca Florea and Viktoria Stray. 2019. The skills that employers look for in software testers. *Software Quality Journal* 27 (2019), 1449–-1479.

[14] Gizem Hekim. 2022. The Best MOOCs and E-Learning Platforms. *Dan Institute* (2022). [Online]. Available at: https://daninstitute.com/blog/the-best-moocs-and-e-learning-platforms/.

[15] Khe Foon Hew and Wing Sum Cheung. 2014. Students' and instructors' use of massive open online courses (MOOCs): Motivations and challenges. *Educational Research Review* 12 (2014), 45–58.

[16] Mehdi Jazayeri. 2004. The Education of a Software Engineer. In *Proc. International Conference on Automated Software Engineering (ASE)*. IEEE, xviii–xxvii.

[17] Mohamad Kassab, Phillip Laplante, Joanna Defranco, Valdemar Vicente Graciano Neto, and Giuseppe Destefanis. 2021. Exploring the Profiles of Software Testing Jobs in the United States. *IEEE Access* 9 (2021), 68905–68916.

[18] Kenneth R. Koedinger, Jihee Kim, Julianna Zhuxin Jia, Elizabeth A. McLaughlin, and Norman L. Bier. 2015. Learning is Not a Spectator Sport: Doing is Better than Watching for Learning from a MOOC. In *Proceedings of the Second (2015) ACM Conference on Learning @ Scale*. ACM, 111–120.

[19] Lan Ma, Saeed Pahlevan Sharif, and Kok Wei Khong. 2022. What factors drive the purchase of paid online courses? A systematic literature review. *Journal of Marketing for Higher Education* (08 2022), 1–24.

[20] Chinh Ngo. 2023. 10 Best Free and Affordable Platforms for Online Courses. *BestColleges* (2023). [Online]. Available at: https://www.bestcolleges.com/blog/platforms-for-online-courses/.

[21] Mauro Pezze and Michal Young. 2007. *Software Testing and Analysis: Process, Principles and Techniques*. Wiley.

[22] Wan Rahman, Hazura Zulzalil, Iskandar Ishak, and Ahmad Selamat. 2020. Quality Model for Massive Open Online Course (MOOC) Web Content. *International Journal on Advanced Science, Engineering and Information Technology* 10 (2020), 24.

[23] Sergii Sharov, Serhii Tereshchuk, Andriy Tereshchuk, V.A. Kolmakova, and Nadia Yankova. 2023. Using MOOC to Learn the Python Programming Language. *International Journal of Emerging Technologies in Learning (iJET)* 18 (2023), 17–32.

[24] Bin Zhu and Shiming Zhang. 2013. Curriculum reform and practice of software testing. In *Proceedings of the 2013 the International Conference on Education Technology and Information System (ICETIS 2013)*. Atlantis Press, 846–849.