Final Paper: Discovering Unaligned Expectations in Universities and Industry for New Graduates in Computer Science and Software Engineering and Finding Possible Solutions

Alan Franzoni
Georgia Institute of Technology
Trieste, Italy
alan.franzoni@gatech.edu

Hasti Ghabel
Georgia Institute of Technology
Atlanta, GA
hghabel1@gatech.edu

ABSTRACT

******* Add Abstract HERE ******

INTRODUCTION

There is a widespread agreement that new graduates from computer science and software engineering **do not always possess required skills, abilities or knowledge when joining the tech industry**: a lot of entry-level jobs actually require three years of experience [3]; gaps between Engineering Education, and Practice (what an engineer does in real life) do exist [12]; the software industry presents dissatisfaction in relation to the level of recently graduated professionals [10]; there is considerable room for improvement in what is taught to software students [in relation with job relevance] [8]; many employers find that graduates and sandwich students come to them poorly prepared for the every day problems encountered at the workplace [4].

Some universities and programs even took steps to try and fix this problem in some specific classes by doing all kind of things: from purposely hindering and disrupting the software development processes [4], to adapting and incorporating industry training strategies into a software engineering course [10], to creating and adapting a project-based software engineering course that led the students to face with current, real-world engineering problems [6], and to highlight to students how relevant is having and developing critical soft skills to succeed in projects[2].

At first, we thought that different outcomes could come from different programs, so we explored the difference between Computer Science and Software Engineering programs, but those didn't prove really relevant; the official ACM/IEEE curricula for Computer Science [7] and Software Engineering [1], which many universities base their program on, are somewhat overlapping, and some studies trying to highlight differences in outcomes between CS and SE graduates were mostly inconclusive: a lot of core competencies are shared [9] [11]. And, those recently-updated curricula don't seem to incorporate lessons from the aforementioned efforts.

The acknowledgment of this skill gap and the efforts to train new graduates for the industry go back as far as 1992 [5]. So, **if in a quarter of a century little to nothing changed, what is the real matter**?

CHI'16, May 07–12, 2016, San Jose, CA, USA

ISBN 123-4567-24-567/08/06…$15.00

DOI: http://dx.doi.org/10.475/123_4

RESEARCH QUESTION

We have two sets of questions. In the first set, our main question is: **does the perceived skill gap in fresh graduates exist because the academy is unable to provide a good training, or just because a) the academy is not even trying to do that kind of job, and b) the industry is taking that kind of job for granted, or c) the students think they should be getting something that the university has no intention to provide them with?** Here, we ask what could be the reasons that there exist a gap between students, professors, and industry professionals' expectations. We will look into these questions for both **undergraduate** and **post-graduate** level students that are recently graduated from school. We also ask the question that **how the students' degree can improve the chance of getting hired? Do the graduate-level studies help the students to gain adaptive skills in industry more quickly?**

In the second set of our questions, we are asking **what would be the best solutions that bring the university and industry's objectives closer to each other?**

THE HYPOTHESIS

We think that:

- Most students, when picking their major, have little to no idea what they're going to actually study, and they probably expect to learn mostly about programming and creating applications;

- Most teachers, when designing their courses, think about teaching what they deem useful to achieve the so-called *computational thinking* in their students;

- Most professionals and employers, when hiring fresh graduates, expect they'll be able to immediately and fully carry out whatever real-world task is assigned to them.

So, we provide two sets of hypothesis to approach the research questions.

Hypothesis 1: One of the reasons for the perceived skill gap is that all those that should - in an employer's view - care for learning some skills to be used at work, don't actually have that aim during their education phase. In other words, the misaligned expectations between industry and university causes the skill gap. Those skills have impact on job proficiency and possibilities to get hired. We think that the expectations among four groups differ. These four groups are composed of **undergraduate-level students, post-graduate-level students, educators and school staff, and industry professionals**. The main problem is not only that one side hasn't enough resources or skills to achieve a certain goal, but, rather, that there's a different vision or gaol on what should be done, and different and unaligned *rewards* exist for different groups.

Hypothesis 2: We think that the following hypothesis is that the graduate-level studies, on the contrary, can play an important role on reducing the skill gap between industry and university, and therefore, can be considered as one good resolution for that issue. More specifically, we think that **high-quality online graduate-level programs** such as Georgia Tech OMSCS (Online Master of Science in Computer Science) are quite aligned with industry expectations. These such programs also target many people from all over the world, who can become proficient in their job quickly as well as being active in an academic environment.

RESEARCH METHODOLOGY

It would be great to have one-on-one interviews with a lot of people, and ask them what were their expectations before starting their degree and after getting it, to understand professionals' point of view, and to try to understand what teachers and university staff designing courses is actually trying to do; but, that would involve an important amount of time, both for the researchers and all the people involved.

Still, we think we can leverage modern technology to achieve something useful; we plan to **create a survey** where we ask questions to assess the thoughts of students (post-graduate and undergraduate level), university teachers, and industry professionals to discover what it is, in their opinion, the current goal of university degrees (both Bachelor's degrees and post-graduate degree in Computer Science, Software Engineering or whatever a similar degree is called in one's country), and how that affects job proficiency and chances of being hired. Following this, we would like to find possible solutions on how to reduce the skill gap caused by misaligned expectations. In this step, we plan to provide more open-ended questions, where participants can describe their thoughts in more detail. In our questions, we will also point out graduate-level online programs and ask them what would be the benefits of such programs in resolving the described issue.

In summary, we hope to explain some of of the misaligned expectations among universities and industry. Following that, we would like to describe some possible solutions to resolve this phenomena.

Independent variables:

- Category: CS/SE undergraduate student, CS/SE post-graduate students, CS/SE professor/teacher/university staff, industry professional in the tech/software field;
- Age;
- Country of residence;
- Country where somebody got his/her degree (if any);
- Highest completed educational degree;
- Previous programming skills or actual job experience in the tech/software field before starting university;
- If employed, company size and tech department size;

Dependent variables:

All the variables are about a CS/SE bachelor's/post-graduate's degree fresh graduates.

- Self-reported current goal for bachelor's/post-graduate's degree: what they think that university aims at, right now;
- Self-reported ideal goal for bachelor's/post-graduate's degree: what they wish that university would do, if different from current;
- Self-reported expected GPA relevance for actual job proficiency;
- Self-reported university ranking relevance - how the ranking of a well-known university can impact the job qualification?
- Self-reported expected (or actual, for professionals) time to achieve full job proficiency when entering the industry?
- Self-reported expected time to land the first job for a current fresh graduate?
- Self-reported chances of getting hired for new graduates with no previous working experience (both post-graduate and undergraduate levels);
- Self-reported top skill which they think useful at a job that can't be taught at school (if any);
- Self-reported expected proficiency of graduates versus practitioners with a relevant work experience in the same ballpark (4 years) but without a degree, when getting a job;
- Self-reported expected proficiency of graduates versus practitioners with a relevant work experience in the same ballpark (4 years) but without a degree, after one years of being hired;
- Self-reported possible solutions to reduce the expected skill gap between industry and universities for new graduates;

- Self-reported role of online high-quality graduate level programs in reducing the skill gap - how programs like Georgia Tech OMSCS (Online Master of Science in Computer Science) can help to solve this issue?

Based on item 9th and 10th of *dependent variables*, we try to discover whether the perceived gap could actually be "reversed" - so, whether the graduates think they're up for a slow start when compared with practitioners, but are then able to gain the same (or more) speed.

What we perceive is novel is that **we target all categories at the same time**, so we can put their answers in perspective, and that we will **aim for a larger group than previous studies**; we'd really like to reach 300+ answers to our survey.

RECRUITMENT

Since we would like to get a good sample, we'll create a website, where we describe what we are doing, and we'll provide links to the survey and an easy way to share the website itself; We will put our results and the provided graphs from our data analysis on the website to present our work. If anyone is interested to know the results, they can subscribe to our website and they will get the result explanations at the end of the research period. We will use SurveyMonkey or Google Forms as the tool to put our survey questions in it and collect the results. We might also use social media to get to the numbers of participants we need.

We think this research is relevant, since many of our peers giving us feedback thought it was very interesting, so we'll ask our classmates to share the website as much as they can, and we'll ask each survey-taker to share it as well, especially to categories that are different than himself (e.g. "Share this survey with your professors!")

The main purpose of our website is to easily share the goal and description of our research with many people, provide the survey link in the website, and make it a platform that we can share and present our results with whoever is interested to see the results.

RESULTS

Once we collect the data, we will split our data into two parts. The first part will focus on investigating the expectation gap and the second part is focusing on finding the solution to reduce the skill gap. We will use **quantitative** and **qualitative** analysis to describe the results. We will take the quantitative approach for the first part of our investigations. We will use analytical tools (e.g. Excel or coding) to analyze our data. We'll try to uncover patterns that highlight the relations between some of our variables. We will use R programming to analyze our data. Here, we shortly describe the methods we will use to analyze our survey data:

1. Use R programming
2. Calculate the simple statistics for our variables (Mean, Max, etc.)
3. Create graphs for each of the questions with error bars
4. Create appropriate tables and diagrams to use
5. Provide analytical methods such as t-test and ANOVA to verify relationships and trends in our data

In the second part of our survey data analysis, which is focusing on finding best solutions to the issue, we qualitatively analyze the open-ended questions. This qualitative analysis helps us to understand the motivations and opinions on solving the problem. We will start reviewing the data as soon as the first pieces of data are collected. Collecting the data for this section might need more time, since it is based on participants' interpretation and explanation to each of the questions. This need a thorough review and understanding of their explanations to make sure we are not going toward false positives or false negatives.

****** ADD Results HERE ****

LIMITATIONS

We think our research has several limitations.

First: we didn't get as many respondents as we would have liked. We got about half the respondents we'd have liked to see; we miss statistical power.

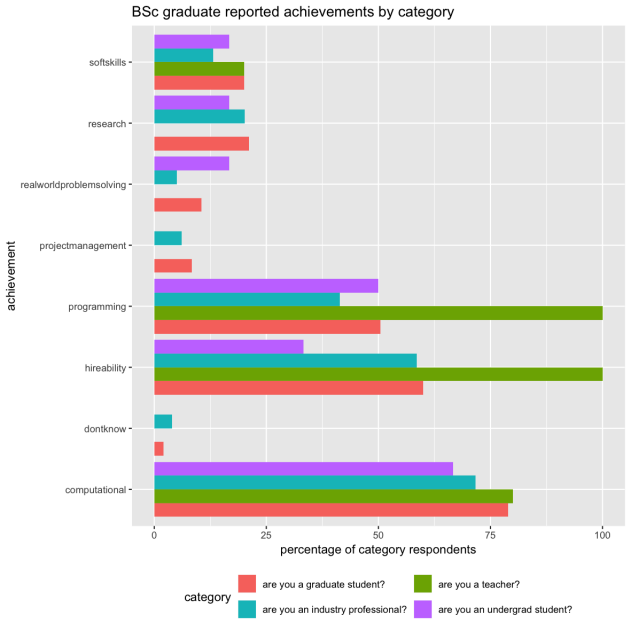
Second: our categories are quite polarized. We have got a lot of industry professionals and graduate students, but we lack teachers and undergraduate students. Surely those few undegrrads and teachers' opinion has a disproportionate effect in our analysis.

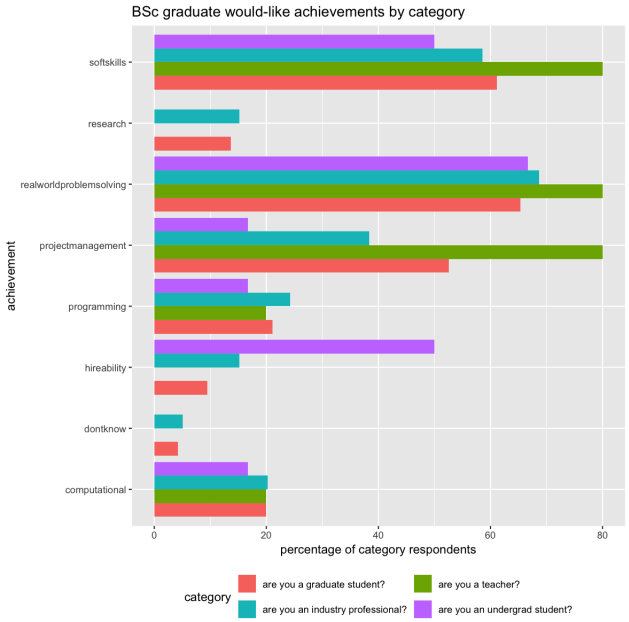
Third: most of our respondents come from the US. We don't think this research can have a worldwide validity, it's probably just one view of the problem.

Fourth: we had initially missed (an error while converting a document to Google Forms) a fundamental question in our survey (the category). Since we knew some of the respondents, we have inferred the categories for our analysis' sake for a small set of initial respondents (the survey was later amended).

CONCLUSIONS

Surprisingly, we've found that the expectations don't seem to be that misaligned on many topics. All of our categories, for example, have got very similar opinions on most BScgraduate skills and would-like skills. With one interesting twist.





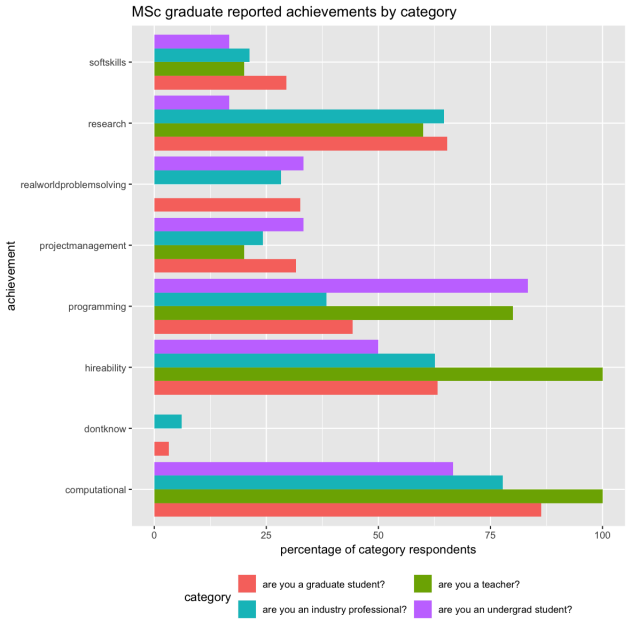
We can see that most of our respondents, regardless of their category, think that a BSc graduate achieves the so-called *computational thinking*, and, secondarily, a certain hireability and some programming skills.

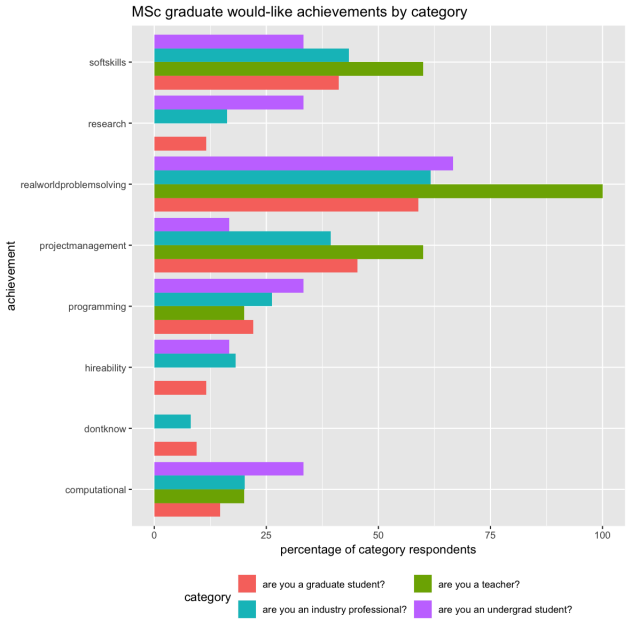
Here, we can actually see a bit of contrast between two categories : the teachers, and the industry professionals. The former think that programming is something a BSc graduate definitely acquires; the latter, being the ones that actually employ such skills, are far more reluctant to say that it's something you get in university.

About the would-like skills, there're not great surprises; it seems that the most interesting that aren't achieved at university we find real-world problem solving, soft skills, and, somewhat, project management - undergrads seem especially uninterested in this latest skills!

But, one surprise comes from programming. It's not marked as a clear achievement but most respondent, and **neither it is marked as a desiderata**. How should somebody hone his programming skills?

The other surprise comes from research skills. Teachers don't think it's something that is taught, and neither that it should be taught. Possibly, that's left to graduate education?





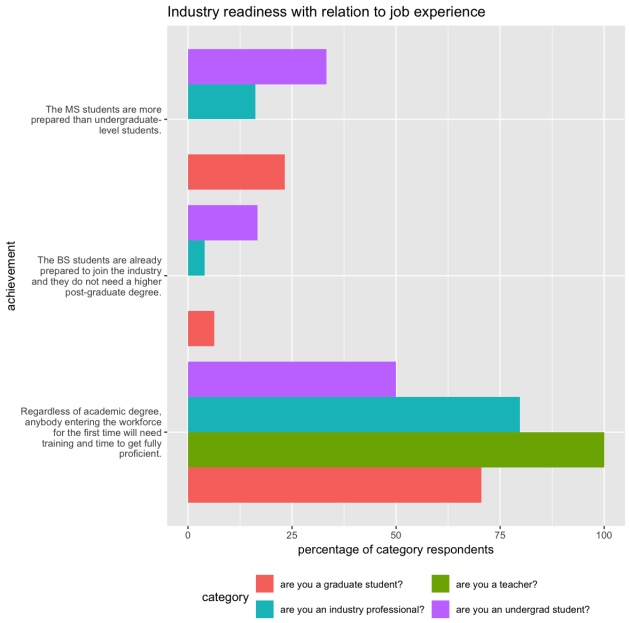
Here we see the opinions about graduate education. It appears to be a partial solution for some parts of undergrad education... but not completely. Most people *but industry professionals* would agree that an MSc improves your programming skills;

since industry professionals are actually the most qualified at such opinion, it's not really encouraging, since they estimate MSc graduates abilities on par with BSc abilities. Programming is hard!

We see better results for research: a reasonable amount of respondents think that graduate education is the stage at which research skills are taught.

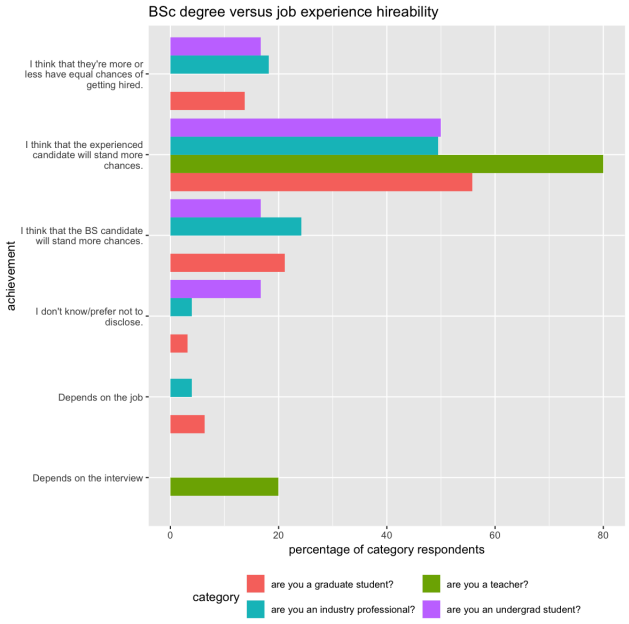
But real-word problem solving is still an open problem, even after a Master's; and soft skills, along project management, could fare much better; there seem to be a large amount of teachers that agree on the fact that Master's graduates would need more skills that can be applied directly in the real world.

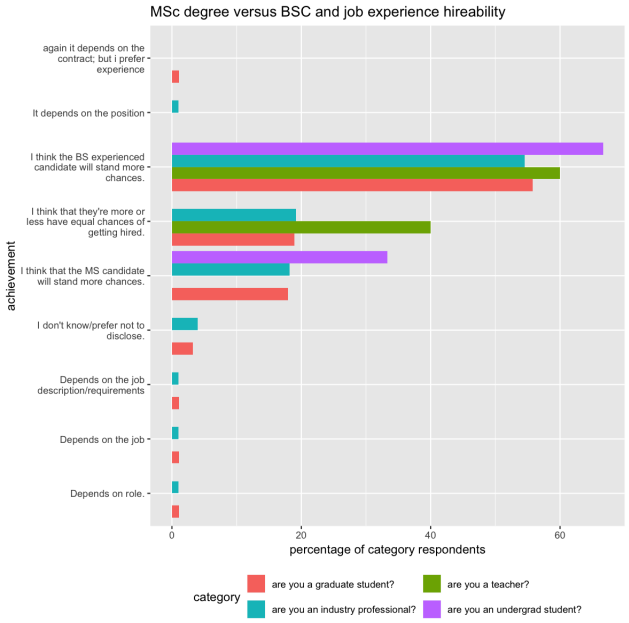
We can see more traces of this belief in another answer:



Only a few undergrads hope that, at the end of their four-year journey, they'll be able to enter the workforce and just have a great career. Other categories mostly believe that an apprenticeship phase will be mandatory for everybody - in this phase, most probably our graduates will learn about **soft skills, project management, real-world problem solving, and.... actual programming!**.

This idea is reinforced by other answers in our survey; when discussing hireability, we asked our respondents to compare experienced and unexperienced candidates with and without degrees:





Hence, about our two main hypotheses, we could say that:

For the first hypothesis, we could say that collected data *partially supports* it. Undergrad programs aren't meant to teach project management, problem solving, or soft skills, while a lot of people would just love to see Bachelors graduate with such abilities; so, there's a misalignment between the intentions of the industry and of the researchers. Interestingly, most undergrad students seem fully aware of the situation.

Programming, on the contrary, seems a matter of teaching abilities. It seems that schools are unable to create good programmers. But then, most people just seem to think that apprenticeship and on-the-job training are not replaceable by pure education; maybe we should just accept that we're yet unable to abstract away that kind of learning from real-world

experience, and we should scale down students (and employers!) expectations about new graduates: they won't be good programmers without an appropriate on-the-job training.

The graduate education part appears to be a bit more foggy. We supposed that graduate-level programs would better fill the school-industry gap, but we cannot say that this is the case. Most of the industry is not concerned with research, and most industry professionals don't see great programming skills in MSc graduates. For sure, it's a beginning, it's something more than basic BSc education; but, probably, spending the same amount of time on a real job would yield the same results about soft skills, real-world problem solving and project management.

POSSIBLE FUTURE WORKS

We think it would be very interesting to replicate the experiment on a different scale, with slightly different premises. **Recruitment** seems to be the hardest part of our research; it would be interesting to partner with some large organizations (be it companies, conferences, universities) in order to push a (similar) survey to them.

It would be interesting to investigate the concept of good programmers, as well. Do bootcamps/MOOCs/other kind of programs produce better coders than university? Are they on par, but within a shorter timeframe? Could the university improve something?

It may be interesting to check if multiple categories (e.g. industry professionals that are MSc students as well) have significantly different answers from "bare" categories; this was a bit outside the scope of our analysis because of time constraints and because of the polarization of our respondents, but could be quite useful.

REFERENCES

1. Mark Ardis, David Budgen, Gregory W. Hislop, Jeff Offutt, Mark Sebern, and Willem Visser. 2015. SE 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. *Computer* 48, 11 (nov 2015), 106–109. DOI: http://dx.doi.org/10.1109/mc.2015.345
2. Maria Cecilia Bastarrica, Daniel Perovich, and Maira Marques Samary. 2017. 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. DOI: http://dx.doi.org/10.1109/icse-seet.2017.15
3. K. Chakrabarti. 2018. 61 percent of Entry-Level Jobs Require 3 Years of Experience. (2018). https://goo.gl/u6wKSc
4. Ray Dawson. 2000. Twenty dirty tricks to train software engineers. In *Proceedings of the 22nd international conference on Software engineering - ICSE '00*. ACM Press. DOI:http://dx.doi.org/10.1145/337180.337204
5. R.J. Dawson, R.W. Newsham, and R.S. Kerridge. 1992. Introducing new software engineering graduates to the 'real world' at the GPT company. *Software Engineering Journal* 7, 3 (1992), 171. DOI: http://dx.doi.org/10.1049/sej.1992.0018
6. David Delgado, Alejandro Velasco, Jairo Aponte, and Andrian Marcus. 2017. Evolving a Project-Based Software Engineering Course: A Case Study. In *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEE&T)*. IEEE. DOI: http://dx.doi.org/10.1109/cseet.2017.22
7. ACM Computing Curricula Task Force (Ed.). 2013. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. ACM, Inc. DOI: http://dx.doi.org/10.1145/2534860
8. Lethbridge. A survey of the relevance of computer science and software engineering education. In *Proceedings 11th Conference on Software Engineering Education*. IEEE Comput. Soc. DOI: http://dx.doi.org/10.1109/csee.1998.658300
9. F. Meziane and S. Vadera. 2004. A comparison of computer science and software engineering programmes in English universities. In *17th Conference on Software Engineering Education and Training, 2004. Proceedings.* IEEE. DOI: http://dx.doi.org/10.1109/csee.2004.1276512
10. Carlos Portela, Alexandre Vasconcelos, Sandro Oliveira, and Mauricio Souza. 2017. The Use of Industry Training Strategies in a Software Engineering Course: An Experience Report. In *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEE&T)*. IEEE. DOI:http://dx.doi.org/10.1109/cseet.2017.16
11. Ghulam Rasool and Touseef Tahir. 2014. A Comparison of Software Engineering and Computer Science Undergraduate Programs in Pakistan. In *2014 12th International Conference on Frontiers of Information Technology*. IEEE. DOI: http://dx.doi.org/10.1109/fit.2014.27
12. Vijayakumar Sivanesan. Analyzing the Gaps between Engineering Education and Practice. (????). https://gatech.instructure.com/files/638493/download?download_frd=1 EdTech Spring 2017.