Proposal: Unaligned Expectations in Universities and Industry for New Graduates in Computer Science and Software Engineering

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

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

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 [14]; the software industry presents dissatisfaction in relation to the level of recently graduated professionals [12]; there is considerable room for improvement in what is taught to software students [in relation with job relevance] [9]; 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 [12], 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 [11] [13]. 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**?

THE HYPOTHESIS

So, we started thinking that maybe the problem is not one of implementation; it's not the university is unable to train graduates for the industry. We began thinking that, possibly, there is a **misalignment in incentives and expectations** for the various stakeholders: industry practitioners, university teachers, and students.

So, the phenomenal question is: what are the expectations and motivations for students, teachers in the field of computer science and software engineering? Following that, what do employers expect from new graduates in this field? Is there any gap between the expectation of these groups?

We have found some research on the topic: students usually do not have a personal vision for what they hope to do with a Computer Science degree, and there's a mismatch between what they are taught and what they had expected [8]; some initial work on understanding what makes a good software engineer was performed [10], but just on a few small samples from a very specific audience.

So, we would like to research into **what the expectations and motivations are** for our groups; we think we'll find a substantial **unalignment** between different groups' aims, that would justify the root reason for the perceived skill gap. So, we think that, most probably, employers will say they'll expect new graduates to be able to perform most real-world tasks; possibly, some students will say the same; but we suspect that most teachers won't answer this way, and they'll just say that they want their students to learn "computational thinking" or be prepared for doing research.

We think, as well, that employers still find some usefulness in new CS/SE graduates, or they wouldn't hire them, and possibly students would not enter such kind of classes; we may able to infer what's the actual usefulness of somebody's studies when doing their jobs. Would it be a good idea to remove

some course which is currently taken in order to improve the immediate perception of readiness for the industry, or would that lead to issues with skills later on?

By the way, we seek mostly a **descriptive** approach to the problem, so we won't cry if the results don't turn out as we expect; we'll have verified a doubt we had.

WHAT WE PLAN TO DO

We plan to **create a survey** where we ask questions to assess the thoughts of students, university teachers, and industry practitioners about:

- What should be the expected skills of fresh graduates? (job proficiency, research proficiency, software writing, computational thinking, etc)
- What are the actual skills they encounter "in the wild"?
- Whether they think gaps exist and should have been filled by somebody else?

The survey will actually be tweaked depending on the target; some questions will be required for some groups, but not for others.

What we perceive is novel is that **we target all groups 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.

In order to achieve such target, 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 hope that a bit of advertising/social media push will help us getting to the numbers we need.

Then, we'll collect the data, graph and analyze it, as well as creating a final presentation to sum up what we'll find.

Our research will be a mix of survey-based and qualitative research. Some questions will be multiple-choice once, but some will be open-answer questions that we'll summarize later on.

The independent variables will be the target groups. The dependent variables will be the groups' expectations and motivations.

LIMITATIONS

Threats to validity, both internal and external: we hope to get a good sample from around the world, but there's always the risk we incur in some biases - e.g. if some universities or organizations get really involved with the research we may get too many results from those specific institutions. We may also get a skewed number of answers from a specific target group - we'll need to understand what a good proportion is between groups as well, because we doubt the world offers the same amount of professionals, teachers, and students.

Additionally, there will be a limitation on finding a large number of participants that meet our group categories. It is more challenging to find a large number of teachers, professors, and also recruiters, since they have a busy schedule or they may not be allowed to participate in certain surveys due to their company policies.

Another thing is time limitation. We only have 4-5 weeks to provide the survey questions, send it to the people (small and large group), validate the questions, and analyze and summerize the results. Sometimes, it takes **a couple of weeks** for each participant to only complete the survey and collect the results.

DELIVERABLES

We collect the deliverables and their due dates in the table below.

Deliverable	Due Dates (year 2018)
Five weekly status reports	Every week, 06/25 - 07/23
Intermediate milestone 1	07/2
Intermediate milestone 2	07/16
Final presentation	07/30
Final project	07/30

Table 1. Schedule of each deliverable

TASKS

Below, we listed the tasks that needs to be completed for this project. We displayed the number of hours for each task in front of it inside the parenthesis.

Milestone 1

Task 0: Collaborate with teammates and complete the **project proposal**. (15 h)

Task 1: Create a **website** that explains our project objectives to the readers and links to our different surveys. This website is required to enlarge the audience with different experiences in the field of computer science and software engineering. (15 h)

Task 2: Plan the **survey questions** that target different categories including students, teachers and professors, employers, and recruiters.The survey questions are very important to lead the research to its goal. Correspondingly, the given answers from different categories can provide a good and promising results analyses. (12 h)

Task 3: Using **proper tools** to provide the survey. Some available tools that we consider are Google Docs, or Monkey surveys. (8 h)

Task 4: Make the **survey** available to a smaller sample of categories. This gives us the opportunity to better understand what we missed in our survey or what was extra, which does not give us important information. We are planning to share the survey at this level with our classmates at Educational Technology - Summer 2018. We will use Piazza platform to share our survey with thisgroup of people. (10 h)

Milestone 2

Task 5: The next step is to **validate** and **update** the survey questions based on the result and feedback from small group survey. (18 h)

Task 6: Create a plan on how to send our results to the people who are interested to know what we find in our research. We can setup a **mailchimp mailing list**. (10 h)

Task 7: Share the **validated survey with a large group of people**. We need to share it on different forums, blogs, linkedIn, facebook, OMSCS slack channel, or maybe targeting employers in some companies. (15 h)

Task 8: **Collect the results** from our survey and describe the results. We will do research on available tools that we can analyze our results and display them on a graph to effectively describe our findings. (20 h)

Final Steps
Task 9: Put all our findings together and **prepare the final project paper**.

Task 10: Complete the **final project** and **final project presentation**. (20 h)

WEEKLY MILESTONES

Here, we describe the weekly due dates, goals, project-related assignments, and associated tasks due dates.

Week #5

- June 11, 2018 - June 18, 2018:
  - **Goal**: Complete the project proposal.
  - **Assignments Due**: Project Proposal
  - **Tasks**: Task 0

Week #6

- June 18, 2018 - June 25, 2018:
  - **Goal**: Provide the website to store the surveys, Provide survey questions.
  - **Assignments Due**: Weekly Status Check 1
  - **Tasks**: Task 1 & Task 2

Week #7

- June 25, 2018 - July 2, 2018:
  - **Goal**: Create the online survey on best possible platform.
  - **Assignments Due**: Weekly Status Check 2, Intermediate Milestone 1
  - **Tasks**: Task 3 & Task 4

Week #8

- July 2, 2018 - July 9, 2018:
  - **Goal**: Validate survey and find solutions on automatically share results with people.
  - **Assignments Due**: Weekly Status Check 3
  - **Tasks**: Task 5 & Task 6

Week #9

- July 9, 2018 - July 16, 2018:
  - **Goal**: Send the online survey for the larger group of people and Collect data.
  - **Assignments Due**: Weekly Status Check 4, Intermediate Milestone 2
  - **Tasks**: Task 7 & Task 8

Week #10

- July 16, 2018 - July 23, 2018:
  - **Goal**: Continue on collecting results, summerize and analyze the collected results to prepare them for the final paper.
  - **Assignments Due**: Weekly Status Check 5
  - **Tasks**: Task 8 & Task 9

Week #11

- July 23, 2018 - July 30, 2018:
  - **Goal**: Complete the Final Project and Final Presentation
  - **Assignments Due**: Final Project, Final Paper, Final Presentation
  - **Tasks**: Task 9 & Task 10

MEMBERS OF THE TEAM

The name and GT username of the team members are mentioned below:

1. Alan Franzoni : *afranzoni3*
2. Hasti Ghabel : *hghabelzadeh3*

TASK DIVISION

The tasks are divided among the team members. The number of each task is described in front of the name of each team member.

1. Alan Franzoni Task #s: 0, 1, 2, 5, 6, 7, 8, 9, 10
2. Hasti Ghabel Task #s: 0, 2, 3, 4, 5, 7, 8, 9, 10

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. Kushal Chakrabarti. 61 percent of Entry-Level Jobs Require 3 Years of Experience. (????). https://talent.works/blog/2018/03/28/the-science-of-the-job-search-part-iii-61-of-entry-level-jobs-require-3-years-of-experience/
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. Michael Hewner and Mark Guzdial. 2011. How CS majors select a specialization. In *Proceedings of the seventh international workshop on Computing education research - ICER '11*. ACM Press. DOI: http://dx.doi.org/10.1145/2016911.2016916
9. 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
10. Paul Luo Li, Andrew J. Ko, and Jiamin Zhu. 2015. What Makes a Great Software Engineer?. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*. IEEE. DOI: http://dx.doi.org/10.1109/icse.2015.335
11. 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
12. 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
13. 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
14. Vijayakumar Sivanesan. Analyzing the Gaps between Engineering Education and Practice. (????). https://gatech.instructure.com/files/638493/download?download_frd=1 EdTech Spring 2017.