Non-cognitive abilities of exceptional software engineers: a Delphi study

Wouter Groeneveld
KU Leuven
Leuven, Belgium
wouter.groeneveld@kuleuven.be

Hans Jacobs
Prato
Hasselt, Belgium
hans.jacobs@prato.be

Joost Vennekens
KU Leuven
Leuven, Belgium
joost.vennekens@kuleuven.be

Kris Aerts
KU Leuven
Leuven, Belgium
kris.aerts@kuleuven.be

Abstract

Important building blocks of software engineering concepts are without a doubt technical. During the last decade, research and practical interest for non-technicalities has grown, revealing the building blocks to be various skills and abilities beside pure technical knowledge. Multiple attempts to categorise these blocks have been made, but so far little international studies have been performed that identify skills by asking experts from both the industrial and academic world: *which abilities are needed for a developer to excel in the software engineering industry*? To answer this question, we performed a Delphi study, inviting 36 experts from 11 different countries world-wide, affiliated with 21 internationally renowned institutions. This study presents the 55 identified and ranked skills as classified in four major areas: communicative skills (empathy, actively listening, etc.), collaborative skills (sharing responsibility, learning from each other, etc.), problem solving skills (verifying assumptions, solution-oriented thinking, etc.), and personal skills (curiosity, being open to ideas, etc.), of which a comparison has been made between opinions of technical experts, business experts, and academics. We hope this work inspires educators and practitioners to adjust their training programs, mitigating the gap between the industry and the academic world.

1 INTRODUCTION

Knowledge of software development is becoming more and more important, as shortcomings in the software engineering workforce require companies to come up with creative solutions such as coding boot-camps, to initiate candidates into the wonderful world of programming. However, to be successful as a developer, it no longer suffices to be technically proficient (Garousi et al., 2019). There is still no agreement on what separates a great developer from a good one, even if both the academic and industrial world are starting to acknowledge the need for something more besides cognitive knowledge, however good this might be (Capretz et al., 2017). Software is first and foremost created by people, for people, hinting on the need for so-called '*soft skills*', or, more generally, '*non-cognitive abilities*', defined as the subset of abilities not related to technical skills.

In a previous literature study, we identified which non-cognitive abilities are perceived as important to educators, and how these are currently being taught (Groeneveld et al., 2019). To complement these findings and to shed more light on possible important skills missed by the review, a Delphi study was performed, gathering data from software engineering experts within both the academic and industrial world. With this study, we aimed to identify and rank abilities besides the obvious cognitive knowledge, perceived as a requirement for people concerned with the technical facets of the software development process. We asked ourselves, and therefore the experts who are more suitable to answer this, the following questions:

- `Q1`: *Which non-cognitive abilities are needed for a developer, to excel in the software engineering industry?*
- `Q2`: *What is the relative importance of these abilities?*
- `Q3`: *Does the opinion of industry experts differ from academics, and if so, in what way?*

The remainder of this paper is divided into the following sections. Section 2 describes related work, while section 3 clarifies the process we have utilized. Next, in section 4 and 5, we present and discuss the results, followed by possible limitations of the study in section 6. The last section, part 7, concludes this work.

2 RELATED WORK

'*What makes a great software engineer?*' is the central question Li et al. answer by interviewing engineers at Microsoft (Li et al., 2015). Li identified a set of 53 general attributes of great engineers, categorized into '*personal characteristics*' (improving, passionate, open-minded, etc.), '*decision making*' (knowledge-oriented), '*teammates*' (mentoring, asking for help, honest, etc.), and '*software product*' (elegant, creative, evolving, etc.). Even if the interviewed engineers come from different divisions, the identified attributes still originate from only one company. While the diverse set of attributes certainly help to understand what affects the success of software projects, the lack of prioritization renders it difficult to translate this list into concrete recommendations for education.

A lot of similar investigations exist in literature, but few seem to primarily target non-cognitive abilities. Some researchers look for industrial requirements by analyzing job ads, risking the inclusion of buzz-words and absence of implicitly required skills (Stevens and Norman, 2016). Ad content analysis and surveys are popular tools used to research trends and identify (IT) skills. However, Delphi studies usually offer more depth in terms of identification of a single concept, such as critical skills for managing IT projects (Keil et al., 2013), or soft competencies in requirements engineering (Holtkamp et al., 2015). Our aim with this study is the same, focusing instead on modern software development.

Another Delphi study was conducted by Wynekoop et al. in 2000, to identify the traits of top performing software developers. This study revealed the most important high-level traits to be geared towards cognitivism (Wynekoop and Walz, 2000). Of course, within a period of 20 years, the software development world has evolved considerably, as the results of this study indicate.

Groeneveld et al. reviewed literature on teaching non-technical skills in software engineering education (Groeneveld et al., 2019). They defined communication, teamwork/dynamics, self-reflection, and conflict resolution as most popular skills to teach students, while creativity, ethics, and empathy dangle at the bottom of the priority list.

So far, we have not found a study that includes both the opinion of people from the industry and people from the academic world. It will be interesting to see what both parties have to say. As is the case with most related studies, high-level discovered traits such as '*good interpersonal skills*', '*team oriented*', '*leadership skills*', and '*technical proficiency*' lack depth, serving as a whole category rather than an individual trait. To mitigate this shortcoming, we made sure to break down general abilities into multiple specific low-level ones, as revealed in section 3.

3 METHODOLOGY

To be able to answer our research questions without merely relying on analysis of existing data, we needed the help from industry veterans. Since simply identifying skills would not sufficiently answer our questions, we opted for the Delphi approach, of which we have applied the most conventional implementation, collecting opinions of different groups called *panels*. The Delphi technique is an effective tool to reach a group consensus on a given topic, where the opinion of experts is used for decision making (Okoli and Pawlowski, 2004).

As stated by Okoli et al., expert selection is one of the most important aspects in a Delphi study (Okoli and Pawlowski, 2004). We adhered to their proposed participant selection model by identifying relevant stakeholders, carefully reviewing their expertise based on information at hand, such as blogs, CVs, social media, and papers, categorizing the identified experts into separate stakeholder panels, and inviting experts until a pre-set limit of 10 to 18 members per panel is reached.

Participating in a Delphi study requires a hefty commitment, since there are several rounds to go through, with the first open-ended round being the most cognitive demanding. Because of that, the availability of information regarding the process and a correct formulation of questions were given special attention. Potential panel members were contacted and asked to participate to reduce drop-out rates during the study.

We invited experts from three different fields, as visible in Table 1: (T) *technical experts* (software developers, software architects, technical coaches, etc), (B) *business experts* (development and HR managers, agile coaches, analysts, etc), and (A) *academics*. The first two groups consist of people in daily close contact with software engineering, while the last group contains researchers and/or educators with experience in the same field.

Our aim at answering the research questions was to deliberately not limit ourselves to collecting responses from domestic software engineers, but to also include opinions of as many widely recognized experts as possible from well-known international companies. Hence the choice of distributing the survey through email. This choice comes with the unfortunate downside of initial low response rates combined with a possible drop-out between rounds.

For this study, the Qualtrics web-based survey system was used. Information was gathered as follows: starting with an open-ended questionnaire (1), after data processing and grouping, we ask participants to verify and potentially revise their answers (2), continuing with the selection of the top x most important skills (3),

Table 1: Panel groups, response rates, and mean duration in minutes for each phase.

Round	Dur.	Panel T	Panel B	Panel A
Invites		35	31	80
Members		14 (40%)	12 (39%)	10 (12%)
#1	35m	11 (79%)	11 (92%)	10 (100%)
#2	6m	11 (79%)	8 (67%)	7 (70%)
#3	6m	9 (64%)	10 (83%)	9 (90%)
#4.1	10m	9 (64%)	9 (75%)	8 (80%)
#4.2	6m	9 (64%)	8 (67%)	9 (90%)

to finish the survey with a series of ranking iterations (4), until a consensus is reached or two iterations have passed. We will briefly discuss each step below, as shown in Figure 1.

Figure 1: Delphi survey structure.

1. Identification of skills: The first questionnaire is the most crucial one, designed as an open-ended digital brainstorming session. Participants are asked to answer our first question by listing at least *five abilities* they think are important. Because we expected this to be answered with typical high-level keywords such as 'communication', 'teamwork', and 'collaboration', we asked two more questions. First, respondents were required to include a brief explanation (2 to 3 sentences) of *why* these are important to them. Next, they had to imagine a developer (possibly themselves) working with a colleague on a problem, specifying *what behaviour* they observe that shows this colleague has the qualities answered in the first question. This extra data made it possible for us to move from a general keyword to a specific ability. In order to avoid confusion regarding the terms used in the questions, we provided a definition of 'abilities', 'non-cognitive abilities', and 'developers', as consultable at `https://people.cs.kuleuven.be/~wouter.groeneveld/delphi`.

2. Verification of results: To ensure the correctness of coding the open responses from round 1, we recruited the help of a seasoned software developer (second author), who was familiar with the problem at hand, but did not participate in the survey. He analyzed two thirds of the anonymized raw data, creating his own set of categories. After comparing and correcting the results, a final run through all the responses resulted in the finalized list of identified abilities. After this coding step, the next questionnaire asks respondents to identify their answers, and if needed, provide further suggestions. This step in the Delphi process verifies the correctness of our coding system, and gives panel members a chance to add to their previous answer. When discussing the results between authors, duplicates were eliminated, so that the original wording of the respondents was retained as much as possible.

3. Selecting most important skills: Now that the complete list of abilities has been identified, we asked participants to select *at least 10 factors* they consider to be the most important skills for a developer to excel in the software engineering industry. In this round, categories have been removed, and the order of all choices has been fully randomized. It is important to notice that from round three and up, the list of abilities will be narrowed down and ranked from within each panel, while the first two rounds were conducted jointly by all panels.

4. Ranking skills: To reduce the ordered skill set from round 3 into a manageable size for each panel, only selected skills by at least one member of this panel will remain for respondents to rank by importance, using a drag-and-drop interface. The last round has to be repeated until a general consensus is reached, measured by calculating *Kendall's* W `>= 0.7` (`1` = complete agreement, `0` = no agreement) (Okoli and Pawlowski, 2004). If W does not rise to the desired level within two iterations, the process will stop anyway, as to not overburden panel members with questionnaires. The end result of the Delphi study nets us three separate lists of ranked, important non-cognitive attributes, identified by each group of stakeholders.

The results in between rounds were not fed back to the participants, except for the data needed to compose the next round. Everyone received a summary of their own answer after each round. Panel members did not know each other, but their identity was visible for the researchers. Data from drop-outs was not excluded from subsequent rounds, and no additional people were invited once the study started.

4 RESULTS

Panel members completed the survey from 11 different countries world-wide, with 17 members from Belgium. Members came from 12 different companies and 9 different learning institutions, including our own university. 146 people have been invited to take part, with 36 positive responses, leading to an invite response rate of 25%, with 39% from the industry and merely 12% from academia, of which 89% participants successfully completed the first round.

Several well-known authorities in the field of enterprise software development took part, as well as (co-)authors of praised books. We made sure not to invite more than four experts from the same company or university, reducing the risk of creating homogeneous groups. Companies involved differed in size, from small to internationally renowned. Thus, we believe panel members to have accumulated sufficient experience to be able to contribute to this research.

Table 2 contains the full list of identified abilities, including a final ranking for each expert panel. Our aim was to code the skills as detailed as possible (e.g. *adjust your communication to the target audience*, *visually communicate ideas*, *speak well in front of an audience*) instead of listing high-level skills (e.g. *communication*, *presentation skills*, *leadership*). Skill categories closely related with cognition (e.g. *intelligence*, *critical* and *analytical thinking*) were also split up, while too technical answers were omitted from subsequent rounds, as these are not relevant to answer the research questions.

There was very little consensus within each panel during the first ranking round: W was calculated at `0.16`, `0.12`, and `0.16` for panel T, B, and A, respectively. After rearranging skills according to the new mean ranks and providing a summarized motivation as feedback, the second ranking round reached a much higher agreement: W reached `0.74`, `0.80`, and `0.58`, proving to be sufficient to end the survey.

5 DISCUSSION

Some abilities are closely related (e.g. *sharing knowledge* and *coaching others*), but still sufficiently dissimilar to justify separation. However, some participants showed difficulties understanding these subtle contrasts while prioritizing and ranking them in round 3 and 4, contributing to a low consensus rate. We felt that digital surveys did not offer enough room for discussion between panel members to clear out any disagreements on precisely defining these terms, despite our attempts to answer questions and disseminate information between rounds.

In general, we noticed a clear pattern in the answers, that is reflected in two groups of two categories: communicative skills (1) & collaborative skills (2), and problem solving skills (3) & personal skills (4). As one participant rightfully observes:

> '*To me, software engineering is a balancing act between being able to communicate with other people and solving problems.*'

The first group focuses on getting along, while the last two categories contain problem solving skills and engineering inventiveness. We will briefly discuss each category, highlighting interesting findings, continuing with a look at the degree of consensus for each expert panel.

5.1 Communicative Skills

As mentioned in (Groeneveld et al., 2019), terms such as 'communication' and 'teamwork' are most common when investigating skills to teach students. It is therefore no surprise that abilities such as *understanding and engaging with people involved*, *adjusting communication to the target audience*, and *actively listening* are reflected in the results. It is interesting to note that technical experts selected 5 out of 11 communicative skills to be of great importance, while business experts only selected 2, and academics 4. *Having the courage to raise concerns* was only picked by panel T and *understanding and engaging with the people involved* only by panel A. Communication is also seen as a way to structure and guide the flow of information within a group:

> '*Teams usually have to find a way to communicate, to try out several ideas. Communication also requires structure: a common system to make notes and have weekly/daily meetings to discuss these notes and decide how to continue.*'

Frequently mentioned skills during the first round such as *clearly expressing intentions to avoid misunderstandings* (8 occurrences) and *showing respect by being patient* (6) were not deemed important enough to make it to the final rankings.

	The ability to...	**Occ. #**	T (Occ. round 3)	**T** (Final rank)	B (Occ. round 3)	**B** (Final rank)	A (Occ. round 3)	**A** (Final rank)
Communicative Skills (11)	adjust your communication to the target audience	11	4	16^{th}		/	5	12^{th}
	actively listen	9	6	10^{th}	5	10^{th}	4	6^{th}
	understand and engage with the people involved during development	8		/		/	6	1^{st}
	clearly express intentions to avoid misunderstandings (including in code)	8		/		/		/
	show respect by exercising patience	6		/		/		/
	visually communicate ideas/problems (e.g. drawing on a whiteboard)	4	5	13^{th}	4	13^{th}		/
	empathize with others by understanding others' viewpoints	4	4	7^{th}		/	4	4^{th}
	be aware of, and to a certain extend control, own and others' emotions	2		/		/		/
	have the courage to raise concerns	2	4	15^{th}		/		/
	demonstrate maturity by admitting when you're wrong	2		/		/		/
	speak well in front of an audience	1		/		/		/
Collaborative Skills (16)	collaborate with others to achieve a shared goal	10	4	6^{th}	5	1^{st}	7	2^{nd}
	learn from each other by sharing/evangelizing/teaching/... knowledge	9		/	6	6^{th}		/
	be accountable by showing, taking, and sharing responsibility	8	4	12^{th}	5	4^{th}		/
	give and receive feedback	8	4	17^{th}	6	9^{th}	4	8^{th}
	(pro-actively) ask for help when needed	6		/	5	12^{th}	4	13^{th}
	(pro-actively) help others when they encounter a problem	5		/		/		/
	negotiate a solution that resolves tensions between stakeholders	4		/	5	14^{th}		/
	show and build a sense of belonging to team- and companywide values	3		/		/		/
	mentor/coach/train others by providing guidance	3	4	18^{th}		/	4	15^{th}
	promote well-being by contributing to team spirit	2		/		/		/
	maintain a positive attitude that inspires others	2		/	4	7^{th}		/
	practice discipline by acting according to established team rules	2		/		/		/
	refrain from frequently interrupting others	2		/		/		/
	work autonomously within a group	2		/		/		/
	resolve conflicts by taking advantage of new perspectives	2		/		/		/
	accept diversity in the workplace	1		/		/		/
Problem Solving Skills (13)	show perseverance by not giving up easily and keep on trying again	11	4	11^{th}		/		/
	be creative by approaching a problem from different angles	10	7	3^{rd}	4	8^{th}	4	9^{th}
	show attention to detail	4	5	9^{th}		/		/
	focus on complex problems for longer, uninterrupted periods	3		/		/		/
	systematically verify assumptions and validate results	3	5	1^{st}		/	5	5^{th}
	reframe problems by observing the bigger picture	3	7	8^{th}		/		/
	draw on multiple sources and domains for ideas	2		/		/		/
	create a clean structure leading to predictability and reliability	2		/		/		/
	like the challenge of a difficult problem	1		/		/		/
	think in a solution-oriented way by favoring outcome over ego	1	4	14^{th}	7	2^{nd}		/
	focus on efficiency and re-use instead of reinventing the wheel	1		/		/		/
	measure (own/others/project) progress and improvement	1		/		/		/
	know which actions to take under which circumstances	1		/		/		/
Personal Skills (15)	be curious by wondering why and how something works	10	7	2^{nd}		/	5	10^{th}
	devote oneself to continuous learning	8	8	4^{th}	7	3^{rd}	5	14^{th}
	be open to the ideas of others	7	7	5^{th}	4	5^{th}	6	3^{rd}
	be a fast learner to keep up with the pace of new technologies	7		/		/	4	16^{th}
	be passionate in what you do	5		/		/		/
	criticize oneself and others objectively and fairly	4		/		/	4	11^{th}
	be flexible by quickly adapting to changes during development	3		/		/	5	7^{th}
	be consistent by doing as you say and saying as you do	2		/		/		/
	set goals and stick to them	2		/		/		/
	be adventurous by willing to take risks	1		/		/		/
	carefully choose which new trends to follow and which to ignore	1		/		/		/
	be stress-resilient	1		/		/		/
	to cope with and learn from failure	1		/	4	11^{th}		/
	develop a sense of pride in your work	1		/		/		/
	leverage self-knowledge to continuously improve oneself	1		/		/		/

Table 2: The full list of 55 identified skills, according to technical experts (T), business experts (B), and academics (A), sorted by occurence in round 1. Occurences per panel from round 3 are indicated in grey. More information, including a comparative visualization, is available at `https://people.cs.kuleuven.be/~wouter.groeneveld/delphi`.

5.2 Collaborative Skills

Being a good communicator is not enough: enterprise software development also requires working together, *in close collaboration to achieve a shared goal*, the most popular skill in this category. It is self-evident that communicative skills also work in favor of collaboration. Therefore, some items are interchangeable between this category and the first.

Business experts especially seem to appreciate these skills, deeming 7 out of 16 of them as important in round 3, with *learning from each other by sharing/evangelizing/teaching knowledge*, *negotiating*, and *maintaining a positive attitude* being skills not picked by the other two panels. However, *mentoring/coaching/training others* was indeed picked, and might also be interpreted as 'sharing knowledge'. Skills that promote *belongingness* have been highlighted by several respondents, reporting that software development is a team sport nowadays:

> '*The ability to function in a team, to help with the norming, forming and performing of a team is crucial. In the end it's the team who is committed to the success of the software, so we need developers who are committed to the team.'*

The same holds true for *negotiating*:

> '*Software is usually written for non-technical customers and stakeholders, for commercial reasons. There is always a tension between technical constraints, commercial needs, the needs and desires of different stakeholders etc. Successful software development involves negotiating a solution that best resolves those tensions.*'

However, compared to other items in the result set, these barely made it to the final ranking. *Asking for help when needed* is seen as more important than *refraining from interrupting others*, and *working autonomously* again was given less attention than heavy team-oriented skills.

5.3 Problem Solving Skills

Besides getting along, a second central theme to the skill set of a good software developer is the ability to identify and effectively solve problems. The Software Engineering Body of Skills (SWEBOS) list by Sedelmaier et al. splits this category into *consciousness of problems* and *competence to solve problems* (Sedelmaier and Landes, 2014). Response data also shows that being able to identify a problem is something different than being able to actually solve it. It comes to no surprise that technical experts marked 6 out of 13 skills from this category as important. What did surprise us was the low amount of these skills picked up by other panels.

Showing perseverance, the most popular skill (11 occurrences), was only chosen by panel T, as did *attention to detail* (4), and *reframing problems by observing the bigger picture* (3). We feel that the ability to *focus on complex problems for longer, uninterrupted hours*, although mentioned (3), is underrated, as this kind of 'deep work' or 'flow' has been proven to aid in problem solving (Newport, 2016). When it comes to software development, '*the devil is in the details*', as one respondent said:

> '*When telling a computer what to do, the devil is in the details. I have never known a good programmer who doesn't pay careful attention to the details of the problem, the code, and the process.*'

Many skills, including *creativity*, can be amplified in the presence of other abilities. They are connected in a systemic way that is difficult to identify in a Delphi study, apart from analyzing open ended question responses, such as this one:

> '*The ability to think of new and creative ways to do things is essential for any kind of problem solving. but it is particularly important in software where complexity requires good design. Creativity includes curiosity, questioning, exploring multiple paths, and converging on a design.*'

5.4 Personal Skills

The last category represents characteristic attributes of individuals, as a combination between willingness and the ability to use a skill is required to succeed. These dispositions include *curiosity*, *being open to ideas of others*, *self-criticism*, and intrinsic motivation such as *being passionate*, which are all consistently reported in critical thinking research (Facione, 2000). Some items from the previous category can also be placed within the context of personal attributes.

For academics, this category is the most important one, with 6 out of 16 skills that made it to the final ranking, while other panels only include 3. While *being a fast learner to keep up with the pace of new technologies* (7 occurrences), *criticize oneself and others* (4), and *being flexible by quickly adapting to changes* (3) are popular choices, they failed to stay on target except in panel A. *Coping with and learning from failure* is an ability picked only by business experts. Developers do acknowledge the need for resilience, as one panel member stated:

> '*Things will go wrong, you will make mistakes, you will misunderstand requirements or technology. You need to be able to not just cope with, but learn from that failure.*'

This is also called '*creating a safe haven*' by (Li et al., 2015), where engineers can learn and improve from mistakes without negative consequences.

5.5 Expert panel opinions

Perhaps unsurprisingly, industry and academic experts reached a significant difference in opinion, as also reported by Eskandari et al., in a Delphi study to suggest engineering curriculum enhancements (Eskandari et al., 2007). Figure 2 visualises these differences by comparing the normalized sum of ranked skills as weights for each category.

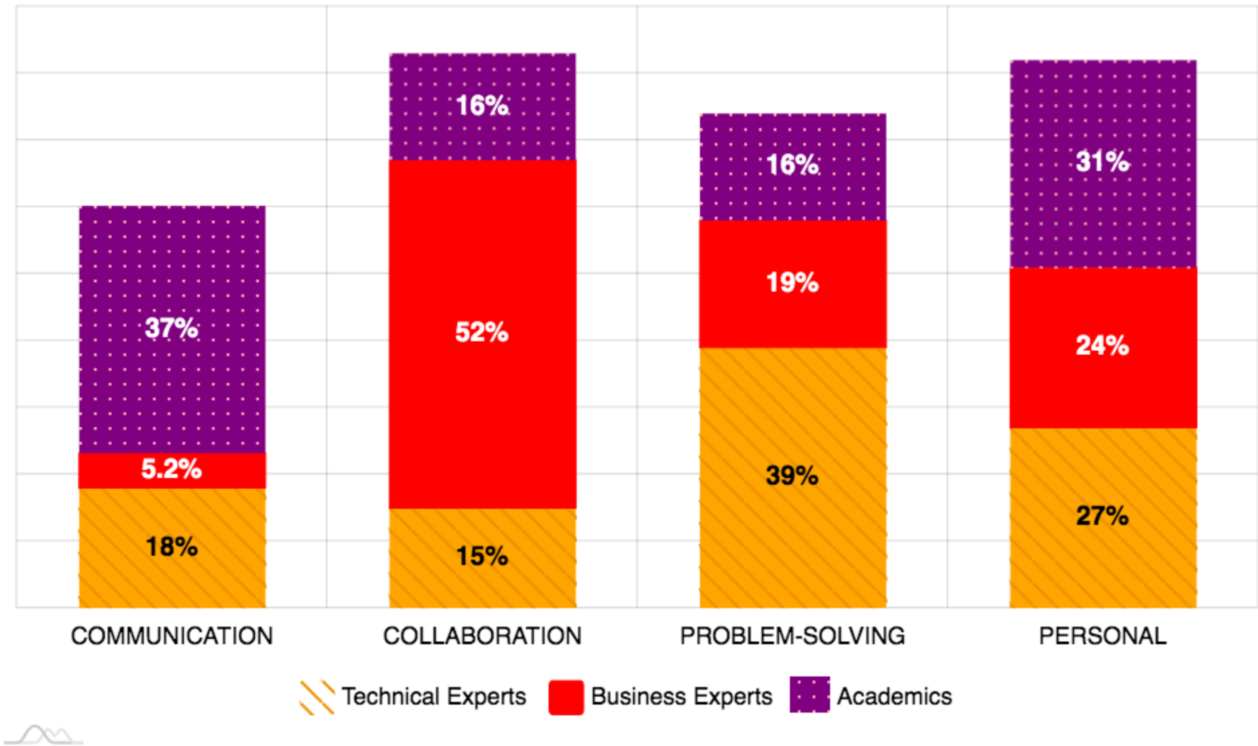

Figure 2: A visual comparison between expert panel opinions, divided in the four skill categories.

Technical experts (panel T) claim *systematically verifying assumptions* (#1) is one of the most important abilities to be successful, together with *curiousness* (#2) and *creativity* (#3). Without *continuously improving* (#4) and learning, any necessary non-cognitive skill becomes unsustainable. Problem solving skills and personal skills matter most.

Business experts (panel B) hold *collaborating to achieve a shared goal* (#1) in high regard, next to *thinking in a solution-oriented way* (#2). After all, the most important deliverable for a software developer is working software, amplified by shared success that may involve some personal compromises (Li et al., 2015). *Continuously improving* (#3) comes in third. There are some similarities between business and technical experts. A good balance between problem solving, personal skills, and collaborative skills is required, with a heavy emphasis on collaboration, as visible in Figure 2.

Academics (panel A) consider *understanding and engaging with people involved* (#1) to be an important aspect, together with *collaborating to achieve a shared goal* (#2) and *being open to the ideas of others* (#3). They conclude that personal skills and communication are of the utmost importance. The difference in opinion about communicative skills between academics (37%) and business experts (5%) is striking.

A possible explanation for the heavy emphasis on problem solving for technical experts could be the tangible and immediate usability, while more abstract personal characteristics that are preferred by academics help with the norming and forming of those practical skills. However, the industry seems to expect ready-to-use practical problem solving skills (Bailey, 2014). This may depend on your professional experience, as one respondent points out:

> '*I found it very difficult to rank the skills by importance. Relative importance can depend on where you are on your career.*'

This is another reason why reaching a certain degree of consensus within each group was not easily achievable.

6 LIMITATIONS

By carefully following the steps explained in section 3, and by comparing our data with existing research, we are convinced that our results can answer `Q1` and `Q3` safely. However, since a Delphi study never reaches any statistical relevance, the answer to `Q2` regarding relative importance of each skill might vary, depending on the formation of the panel. Also, since the results are highly contextual, ranking them might not always be very relevant.

As mentioned before, depending on where you are in your career, certain skills might be more important to you than others. Also, depending on the interpretation of the questions or the abilities themselves, answers might vary. It is very difficult to facilitate a heated debate using only a digital survey.

This could be mitigated by performing additional focus groups using the results from this study as input, or by repeating round 3/4 in Figure 1 with a new set of experts, until the outcome remains steady. Regional and cultural differences might further complicate things. Generalization was, however, never the goal for this research. We believe that our systematic procedure yielded interesting and relevant findings.

So far, we have been using the term 'non-cognitive skills' to identify the group of abilities. We asked participants to come up with a more fitting alternative, given their selection of skills. The answers vary and are listed at our website. While no single answer seems fitting enough to harbor each category and result from Table 2, it is clear to us that 'non-cognitive' is the wrong word, as one participant elaborates:

> '*Non-cognitive seems not only incorrect but derogatory. All non-technical skills are cognitive, require practice, consideration and deliberate exercise to improve.*'

7 CONCLUSION

The 36 panel members of the Delphi study presented in this paper agreed upon important abilities needed for a developer to excel in the software engineering industry. By visualizing and comparing results, we have uncovered interesting differences in opinion between panels. These findings will hopefully help to close the gap between industry and higher education. Future work might dig deeper into understanding these dissimilarities.

The legitimate remark regarding your personal career stance raises the question of which skills are absolutely essential for a starting developer, and which skills will be developed as your experience grows. The skill categories and voting trends visible in Table 2 are more interesting for further investigation than the level of agreement within the ranking rounds. It is important to ask ourselves which skills from the list should be taught to students, and which skills they will develop themselves as practitioners.

To further reconcile industry and academia, our future work will involve zooming in on underexposed abilities that were considered important by either party, and to investigate how to teach and evaluate these abilities in higher education.

8 ACKNOWLEDGEMENTS

We wish to thank all survey respondents for their valuable contributions to this research, who despite the annoying reminder emails and long questionnaires, still managed to find time and effort to participate. Thank you all very much!

References

Janet L Bailey. 2014. Non-technical skills for success in a technical world. *International Journal of Business and Social Science* 5, 4 (2014).

Luiz Fernando Capretz, Fahem Ahmed, and Fabio Queda Bueno da Silva. 2017. Soft sides of software. *arXiv preprint arXiv:1711.07876* (2017).

Hamidreza Eskandari, Serge Sala-Diakanda, Sandra Furterer, Luis Rabelo, Lesia Crumpton-Young, and Kent Williams. 2007. Enhancing the undergraduate industrial engineering curriculum: Defining desired characteristics and emerging topics. *Education+ Training* 49, 1 (2007), 45–55.

Peter A Facione. 2000. The disposition toward critical thinking: Its character, measurement, and relationship to critical thinking skill. *Informal logic* 20, 1 (2000).

Vahid Garousi, Gorkem Giray, Eray Tuzun, Cagatay Catal, and Michael Felderer. 2019. Closing the gap between software engineering education and industrial needs. *IEEE Software* (2019).

Wouter Groeneveld, Joost Vennekens, and Kris Aerts. 2019. Software engineering education beyond the technical: A systematic literature review. In *Proceedings of the 47th Annual SEFI Conference*.

Philipp Holtkamp, Jussi PP Jokinen, and Jan M Pawlowski. 2015. Soft competency requirements in requirements engineering, software design, implementation, and testing. *Journal of Systems and Software* 101 (2015), 136–146.

Mark Keil, Hyung Koo Lee, and Tianjie Deng. 2013. Understanding the most critical skills for managing IT projects: A Delphi study of IT project managers. *Information & management* 50, 7 (2013), 398–414.

Paul Luo Li, Andrew J Ko, and Jiamin Zhu. 2015. What makes a great software engineer?. In *Proceedings of the 37th International Conference on Software Engineering-Volume 1*. IEEE Press, 700–710.

Cal Newport. 2016. *Deep work: Rules for focused success in a distracted world.* Hachette UK.

Chitu Okoli and Suzanne D Pawlowski. 2004. The Delphi method as a research tool: an example, design considerations and applications. *Information & management* 42, 1 (2004), 15–29.

Yvonne Sedelmaier and Dieter Landes. 2014. Software engineering body of skills (SWEBOS). In *Global Engineering Education Conference (EDUCON), 2014 IEEE*. IEEE, 395–401.

Matt Stevens and Richard Norman. 2016. Industry expectations of soft skills in IT graduates: a regional survey. In *Proceedings of the Australasian Computer Science Week Multiconference*. ACM, 13.

Judy L Wynekoop and Diane B Walz. 2000. Investigating traits of top performing software developers. *Information Technology & People* 13, 3 (2000), 186–195.