Universität Stuttgart

Institut für Softwaretechnologie,
Abt. Empirisches Software Engineering

Universitätsstraße 38
D– 70569 Stuttgart

Bachelorarbeit

Software engineering in theory and practice: Experiences of former students of the University of Stuttgart

Felix Biedermann

Studiengang: Softwaretechnik

Prüfer: Prof. Dr. Stefan Wagner

Betreuer: Dr. Daniel Graziotin

begonnen am: 13.01.2021

beendet am: 12.07.2021

0.1 Abstract

Universities have one of the goals to prepare students for their work life. But in the last years, employers, as well as the graduates, have reported a skill gap between acquired and required skills. The University of Stuttgart offers the study topic Softwaretechnik, which contains a mix of theory and practice with concepts and practices of use and in use, in the industry. To balance this, the university requires feedback.
This sample study analyzes how bachelor alumni perceived their early work life with the knowledge they gained from the university. For this, seven graduates were interviewed, and the data then analyzed using Straussian Grounded Theory.
The results show that the graduates felt not prepared due to missing skills and practice, and required a student job to counter this. But they saw the advantage of the degree and reported to have acquired a broad knowledge as well as metaskills helping them in their first job.
The study emphasizes to include additional smaller projects in the Softwaretechnik study and spread awareness among students and graduates for the feedback system.

Contents

List of Figures

List of Tables

List of Listings

1 Introduction

One of the goals of students attending universities is to get prepared for a future professional career. Studies before have shown a gap, between the student rating and expectations of their skill, and their skills rated by employers ([YAN13], [Tym13], [ABCH], [SOTR16], [MM16]). During these, students and employers were questioned in different countries and therefor representing a worldwide topic for universities, when it comes to the employability of their student and later graduates. Studies were conducted asking employers and students but also questioning graduates ([Nil10], [SJ10] and employers([SGN15]) separately. Employers argued, that graduates were lacking in soft skills ([SOTR16]) as well as graduates them self ([Nil10]).
In computer science-related fields, universities are challenged with balancing their offering of theory and research-oriented with practice-oriented knowledge that is of use and still in use. In Softwaretechnik the students attend lectures in which a mix of theory and practice of the disciplines are offered. The knowledge they gain is supposed to prepare and support them for their first and following steps in their professional career.
Studies about the employability of graduates, rather they intend it or not, are feedback for universities to how useful the knowledge, the graduates learned is and how balanced the curriculum is. The University of Stuttgart offers an annual questioning for quality development, which is aimed at all graduates of the university. For the subject area of Informatics, which includes Softwaretechnik, a feedback system was planned but failed due to a lack of participation. Therefore, there is currently no possibility for graduates who studied Softwaretechnik and started with their work life to give specific and direct feedback to the organizers of the core courses they visited. Especially about the early stages of their work life when they came to use and build upon the concept and practices, they learned, under a business environment.
For this work, it was examined how, former Softwaretechnik students of the University of Stuttgart, experienced the first steps of their work life with their gained knowledge.
The goal was to find out how the alumni felt at the early stages of their work life, which topic and concepts from their study time they used and to which extent.
First, the content of mandatory, shared courses related to Softwaretechnik were inspected. These needed to be offered by the Bachelor of Science program by the University of Stuttgart. From the selected courses the core topics and practices were extracted.
A sample study [SF20] was designed in which multiple former Softwartechnik BSc. students of the University of Stuttgart were interviewed. Using the sample study and following the theoretical saturation strategy these three Research Questions were answered:

1. RQ1 How well-prepared did alumni of BSc-Softwaretechnik feel at early stages of their work life?
2. RQ2 Which concepts and practices as offered by the university have been applied in their work life?

3. RQ3 To which extent concepts and practices as offered by the university were found to be helpful in their work life?

The previously extracted core topics and practices were used as a base for the interview questions. Due to the present pandemic, the interviews were all executed via video call. The resulting interviews were then analyzed using the Straussian grounded theory [SRF16].

2 Related Work

The Idea of this study is inspired by “Onboarding inexperienced developers: struggles and perceptions regarding automated testing“ [PKSS17]. It is a Qualitative study using Grounded Theory about the negative view of inexperienced software engineers towards automated testing with 170 surveyed and 22 interviewed practitioners. Focusing on software testing skills, their experience of companies with recent university graduates was explored to answer the following research questions: “Do software companies see problems with the testing skills of new hires¿‘, “What are these problems, exactly¿‘, “How are companies coping with these problems¿‘ and “What is the impact of new hires' lack of testing skills¿‘.
A skill gap between university graduates and industry expectations is recognized by practitioners. This perception could already affect hiring practices. It is found that different and at times, costly strategies are used by practitioners to counter this gap with training and mentoring efforts. It is also found out that an important factor for the felt competitiveness of the students is the acquiring of skills that correspond to the trends of the industry

“Exploring the intersection between software industry and Software Engineering education“ [CJNZ20] is a Systematic Mapping study exploring the teaching of major Software Engineering trends in project courses. Trends can set in the areas, processes and practices, teaching approaches, and the evolution of Software Engineering Trends over time. The study wants to answer the following questions: “To what extent are software engineering Trends presented in software engineering education research¿‘, “How does software engineering education research present the teaching of various SE Trends¿‘, “How do software engineering Trends contribute to literature¿‘ and “Which bibliographical sources primarily publish studies¿‘.
The systematic mapping study involved the searching, screening, and keywording using abstracts of papers with the topics, software industry, and Software engineering education as main elements. After this, a classification scheme was created. It is revealed that a major trend in Software engineering education is Agile Software Development. Trends that are small in the academic environment, but continuously growing, are Software Implementation, Usability and Value, Global Software Engineering and Lean Software Startup.

Which skills Computer Science alumni are missing according to the industry is researched in “ Gaps between industry expectations and the abilities of graduates“ [RW13]. It is a systematic literature review about determining in which areas computer science graduates lack in meeting the expectations of the industry or academia. To answer this, the three following research questions were defined: “Is there empirical evidence of knowledge deficiencies in graduating computer science students¿‘, “What are the most common knowledge deficiencies in students¿‘ and “Are there trends or changes in knowledge deficiencies in students over time¿‘.
The study followed the Systematic literature review process according to [KC07].
31 categories of knowledge deficiencies were identified. Graduated students are lacking in different

areas which include technical abilities, personal skills, and professional qualities.
The most frequently identified were: Written Communication, Oral Communication, Project Management, Software Tools, Testing followed by Teamwork.

“Exploring the gap between the student expectations and the reality of teamwork in undergraduate software engineering group projects“[IF19] is a Qualitative Study using grounded Theory about investigating the expectations of software engineering undergraduate students before and their experiences after a group project. The study included the following research questions: “What expectations do students have in terms of teamwork at the start of a software development group project¿‘, “How do undergraduate students perceive the reality of teamwork after completing a software development group project¿‘ and “How do the expectations match the reality students face when developing software as a team¿‘.
Expectations are compared with the experiences described by the students at the end of the study. Before the project, undergraduates do not relate teamwork to software development but recognize it as a significant part. Afterwards, Issues in the team and its members, the skills required to fulfill the project, the process put in place, and the team environment overall are reported.

A study that is close to this study is “Engineering graduates’ perceptions of how well they were prepared for work in industry“[MMCF05]. The study is a qualitative study with semi-structured interviews about how prepared chemical engineering students felt for working in the industry. The study aims to explore the following issues: Technical challenges, Communication, Teamwork, Management and, Life-long learning.
The interviews were conducted till theoretical saturation was reached, and the data then analyzed using condensation and categorization. Sixteen graduates were interviewed, and it was found that graduates seemed well-prepared facing challenges of work in the industry in technical and non-technical areas of abilities. Though the former students were lacking in exposure to teamwork, management, and practical experience.

“Enhancing individual employability: the perspective of engineering graduates “ [Nil10] is a Qualitative study in which 20 graduates, from a master-level engineering program in information technology in Sweden, are interviewed. The aim is to show the perceptions of the graduates regarding employability. The study shows that formal and technical skills are considered of less and declining importance in employability compared to soft skills and personal attributes. It also shows that the alumni have different educational expectations from current uni practices. It finally suggests focusing less on substantive skills in engineering and more on soft employability skills, with interpersonal skills included.

3 Methodology

3.1 Survey Method

A Sample Study was chosen to be able to reach generalizability over the population of Softwaretechnik Bachelor alumni. The research questions were defined upfront and open-ended, hence the Straussian Grounded Theory [SRF16, Table1] was used to analyze the resulting data. The data came from interviewing the graduates via video conference. The base of the interview were questions, which were defined before the interviews to answer the research questions. Even though these were predefined the live interviewing allowed to react flexibly on potentially new upcoming aspects during the questioning.
The overall study flow can be seen in figure 3.1.

To make sure a sufficient amount of data is collected, the theoretical saturation strategy, as part of the Straussian Grounded Theory, was used. The conducting of the data collection and analysis can be split into 3 sections: Preparing the Interviews, conducting the interview and analyzing the interview. The last two sections were part of a cycle which is shown in figure 3.1. While the preparation was conducted once for all the interviews, the interview and immediate analyzsis of the data followed the theoretical saturation strategy. After each analysis, the results were examined for new findings. If so, the next graduate was interviewed. It also enabled to examine if the interview questions resulted in the expected answers for the research questions and redefine or add new questions if necessary. If no new aspects appeared from the analysis, theoretical saturation was reached and the interviewing process concluded.

3.2 Research Questions

3.2.1 RQ1 How well prepared did alumni of BSc-Softwaretechnik feel at early stages of their work life?

This open-ended question is supposed to bring clarity about how student faced their open life ahead of them. The interesting point here is that it does not ask for how prepared they were but how they felt. The second one leaves room for cause and effect of their feeling towards their career. The effect being the degree of confidence towards their career cause by the education at the University of Stuttgart. So the questions are about the moment directly after their graduation. This moment will never change for the former students, while the opinion of how prepared they were, can vary over time depending on their experience.

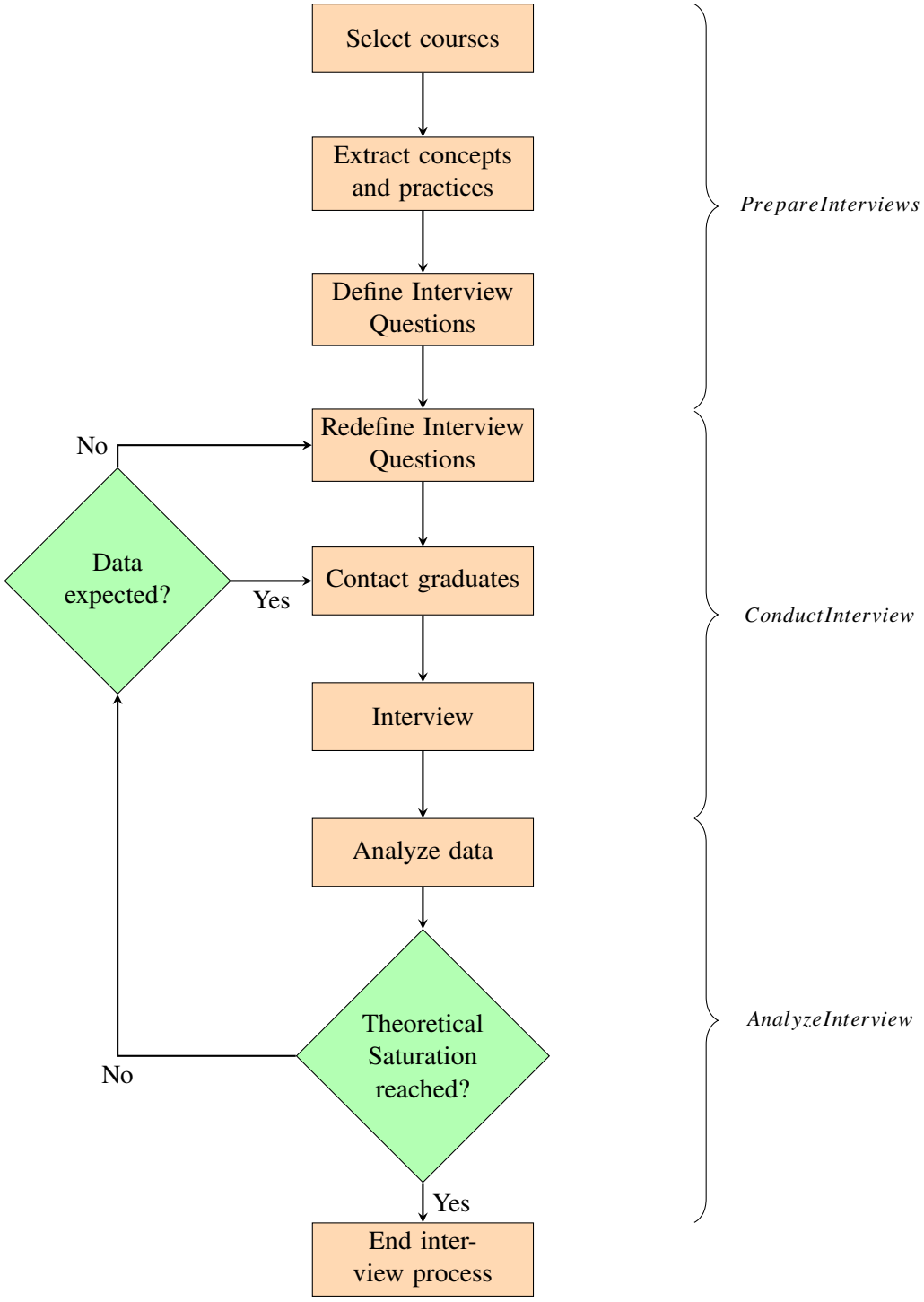


Figure 3.1: Study flow

Listing 3.1 Criteria for selecting core lectures for BSc Softwaretechnik alumni

- Contains concepts and practices which appear in Software Engineering by Jochen Ludewig and Horst Lichter [Joc]
- Have to be completed by Softwaretechnik students
- Contain concepts and practices which overlap

3.2.2 RQ2 Which concepts and practices as offered by the university have been applied in their work life?

The concepts and practices were extracted from lectures which were selected as seen in section 3.3. This was done to be able to go into more detail in those fields. These are supposed to represent a core of the Softwaretechnik specific lectures and help the questioned graduate to remember areas of the learned knowledge. If other lectures, concepts or practices were mentioned they were included in the data as well.

3.2.3 RQ3 To which extent concepts and practices as offered by the university were found to be helpful in their work life?

The previously selected or mentioned concepts and practices were tested for their application in the graduate's career. This included the beginning and the appearance of those, till to the point of the interview. The question is supposed to find out which topics that were taught by the university have or had relevance for supporting the graduates in dealing with tasks in their jobs. The former students also work in different fields, so the relevance of topics can change over time and differ from person to person.

3.3 Preparing Interviews

3.3.1 Select courses

The first step for preparing for the interviews was to find shared mandatory courses related to software engineering which are offered for the BSc Softwaretechnik program at the University of Stuttgart. The Campus-Management-Portal by the University of Stuttgart was used to identify courses that were offered to Softwaretechnik students.
Here courses were chosen that fulfilled conditions as shown in listing 3.1.
The book Software Engineering was chosen as an orientation if a topic is relevant. This decision was made because courses in Software Engineering at the University of Stuttgart are based on the book. It could also appear that the names of courses change over time but because of the base of the lectures, the names of the practices and concepts stayed the same.

Listing 3.2 Criteria for selecting core concepts and practices for BSc Softwaretechnik alumni

- Topic is marked by the professor in an obvious colour to be highlighted
- Topic was part of an exam task in that lecture
- Topic was explained in detail throughout multiple presentation slides
- Serves as a step in the Software development cycle

3.3.2 Extract core concepts and practices

From the selected courses, topics were listed based on indexes and content to serve as a summary. The listed topics were following one or more of the criteria as seen in listing 3.2. These criteria were chosen as they meant, that the professor, teaching those concepts and practices, saw them as particularly relevant for the young software engineers.
After this, the listed topics from the selected courses were compared and marked if they appeared multiple times in different lectures. Topics that were marked were then chosen to be a core concept or practice and to be questioned in RQ2 and RQ3.

3.3.3 Define Interview questions

The questions were defined to be open questions like the research questions and had to be unambiguous according to the guideline for sample studies[SF20]. They were first written in English and then translated into German. The goal was to make the interviewed alumni talk for as long as possible and not run into expression problems. This also could enable the discovery of new aspects of the topic and new material for questions.
The questions were split into four categories: Entrance Questions, RQ1, RQ2, RQ3 and Final questions. Entrance questions served the purpose of leading the graduates to remember the time at university as the name suggests. No specific questions like for lectures or topics appeared. RQ1, RQ2 and RQ3 questions were about answering the respective research question. The final questions were then supposed to end the interview, having the graduate draw a result. Questions to answer a research question could also have sub-questions if they had a common theme.
It was also possible to ask questions not connected to the research questions for additional content. To have enough time for the interview 90 minutes were planned. To confirm this set time frame, a test interview was made with a working student.

3.4 Conducting Interview

3.4.1 Redefine interview questions

Questions could be redefined based on feedback after each interview. Even though they were supposed to be open coding, short answers might appear. Also, questions might be complicated or confusing in their formulation letting the graduate first think about what the questions mean,

Listing 3.3 Strategies to get in contact with BSc. Softwaretechnik Alumni.

- Using OPUS - Publication Server of the University of Stuttgart to find graduates bachelor alumni to be searched on the social network Xing
- Using Xing search engine with the terms Softwaretechnik and Universität Stuttgart to find public profiles
- Searching on the platform LinkedIn using the terms, “Bachelor of Science Softwaretechni“ and “B.Sc. Software engineering“ using the university location “University of Stuttgart“. Found public profiles would be then contacted.
- Using contacts of graduates who replied
- Asking in friend groups for contacts fitting the profile
- Asking supervisor for help in making contact with former students

Name	M. M.
Status	Interviewed
Graduation year	2018
Job	Developer
Date of Interview	30.06.

Table 3.1: Example table for contacted students

making it harder to get into a talking flow. Feedback could be taken into account and redefine or add new questions. New questions could then still be asked to previously interviewed graduates via the platform used for making contact.

3.4.2 Contacting students

To be interviewed were former students of the University of Stuttgart that finished their Bachelor of Softwaretechnik degree and then started their working life. Hence, graduates that completed a master afterwards were excluded from the population to be interviewed. Only graduates which studied between 2011 and 2020 would be contacted to ensure the experience could still be remembered as well as possible.
It was important to use public information to use a privacy-respecting way in contacting them. Therefore if the contact was made through another person the graduate first needed to agree to be contacted.
The following strategies were used to contact graduates as seen in listing 3.3. The message first sends to the former students contained information about the topic of the thesis and the needed time frame. It was kept short on purpose to appear like a text that could be read briefly. When an answer would come, more details could be conveyed. Afterwards, an appointment would be organized for the interview.
Each contacted graduate was written down in a spreadsheet, like in the following example table 3.1.

The table served as preparation for the Interview and for organising who was already contacted and answered. Names were shortened to make them not traceable for privacy protection.

3.4.3 Interview

In advance, the graduates were asked for allowance to record the audio of the interview with the assurance of deleting those files after use. Cisco Webex Meetings was used for the interview itself. Once a date and time were set, a link for the meeting was sent. Webex would also enable the interviewed graduates to use their browser and not having to download another software before the Interview.
During the Interview Action! Screen Recorder by Mirillis was used to record the Audio. The recording was then listened to in the analysis.
For the Scenario of technological problems regarding the recording software, the questions were also printed, so notes could be made during the interview. But the main purpose of the recording was to be able to concentrate on the graduate and react to new appearing aspects or ask further questions.

3.5 Grounded Theory Analysis

For the Grounded Theory approach the Straussian Grounded Theory [SRF16] was used. The steps described in this section were done after each interview till theoretical saturation was reached.

3.5.1 Data Extraction

Each audio file was listened to and the answers of the questions then noted in a spreadsheet. The answers were written in full sentences, written exactly how they were said by the interviewed graduate. The Answer was then translated by using Google Translate and then corrected grammatically if needed. An example can be seen in 3.2

3.5.2 Conducting Coding

Open Coding

The English text was then transformed into code. The code could be a single word describing a lecture, concept or practice. It could also be a short summary of a phenomenon appearing in a sentence[SRF16].
The codes were then sorted into mind maps. One for RQ1 and one for RQ2 and RQ3. As well as one for the entrance and one for the final questions. The last two research questions were combined due to the third one taking the topics of the second and specifying the use of them, building upon them.

Question	Sub question	Answer (German)	Answer (English)	Codes
Entry Questions				
Was hast du nach dem Bachelor in den ersten Schritten des Berufslebens gemacht?		Wir sind zu der Kontakmesse gegangen und paar gute Firmen waren da dabei.Ich hatte dann auch ein Bewerbunggespräch und wurde angenommen.	We went to the contact fair and some good companies were there. I then had an interview and was accepted.	Got job via contact fair
	Wie denkst du haben sich andere in deinen Schuhen bei Ihren ersten Schritten gefühlt?	Die meisten wollten in die Wirtschaft gehen	Most wanted to go into business	Practice over theory after university
RQ1				
...				
RQ2				
...				
RQ3				
...				
Final Questions				
...				

Table 3.2: Table of how data was extracted and sorted

A mind map was chosen as a coding structure to manage the code and sort them into categories. Each coloured node represents a category till the white leaves being the codes. This way relations between categories were visualised and could be created automatically through the process as connections became clear.
The Codes were then put into the Mind Map one by one. The codes were constantly compared [SRF16] through which, a relation between code was realized and therefore added into categories, which described the phenomenon. Subcategories could also emerge due to codes describing a part of the codes in a category. Categories could also reach a size where the overview is at risk, making an analysis harder therefore further subcategories were created. Once a category emerged a note was written which describes what that category contains or describes hence conduction Memoing [SRF16]. Memo sorting [SRF16]was done by using the Mind Map, making the connections between categories and their memos visible. To distinguish selective, axial, normal and subcategories, different bright colours of blue were chosen. This way the existing connection became easier to understand.

Axial Coding

During the open coding processes, connections between categories became clear and became subcategories of an axial category. A relation could also not become clear immediately. Hence, the Mind Map was examined once per day to possibly find new connections. This way the categories were related at a dimensional level.

Selective Coding

Categories previously created in Axial Coding were connected. For some categories, this process happened automatically when new connections became clear. For less obvious connections the mind map was examined by making notes about possible connections. It was first written down what the axial categories represent and which attributes they have. They then were put in context to research questions. Over time a relationship became clear and a node was created representing the selective category.

3.5.3 Analysis

The Mind Maps were examined after each interview. For first analysis was done after the first interview and after the third as well with the following interviews. All categories and their connections were described as well as a summary of the codes in the categories.
With each analysis after the first one, the text was expanded and supplemented as well as corrected when new findings changed a category. Criticism by the graduate as well as ideas that occurred during the interview were taken in account by writing improvement statements for the Redefine step.

3.5.4 Theoretical Saturation

Theoretical Saturation was reached when no new categories were created in the Mind Map, new codes could be sorted into existing categories, creating now new relationships between them. Hence, the analysis did not bring new aspects and the written analysis text stayed the same.

4 Results

4.1 Selected lectures

The courses in table 4.1 were chosen based on the method described in 3.3.

4.2 Extracted Concepts and Practices

The in table 4.2 extracted concepts and practices were selected as described in 3.3.

Course	Reason
Software Qualität (SQ)	For young Softwaretechnik students this course was the first encounter basic concepts and practices of software engineering. During the exercise, they worked with Java and had to use the skills from the lecture like Clean Code, Refactoring, Guidelines and Troubleshooting. that are also covered in the book Software Engineering [Joc]
Einstieg in die Softwaretechnik (EST)	Students learned more about basic terms and techniques in software engineering and software development. These include software development process models, agile approach, Scrum, Specification, software design, implementing and testing. These were also covered by the book Software Engineering [Joc]
Software Engineering (SE)	The lecture build upon Einstieg in die Softwaretechnik and deepens the knowledge in software engineering. The topics, that are based on the Software Engineering Book [Joc], include: Process management, Requirements Engineering, Softwarearchitecture, Debugging, Quality Assurance, Maintenance and Model Driven Software Development.

Table 4.1: Table of chosen courses

Concept and Practice	Appearance	Reason
Vmodel	EST, SE	According to the courses the V-model serves as a base for different software development process models. The goal was to find out if graduates see it the same way and if the term played a role in their work life
Project management	EST, SE	During their study, the students learned about different software development process models. Especially Scrum was taught as an important model for an agile approach
Requirements Engineering	EST, SE	The students experienced and learned about analysis and specification processes. It was done in theory and in practice.
Architecture	SE, EST	Planning and building the software architecture as well as understanding the different layers to it was taught with this topic.
UML	SE	UML was separated from architecture due to its many use cases and appearances in the Softwaretechnik study time in general. It was checked in the exam with a guideline to follow.
Software Quality Assurance	EST, SQ	As part of this topic, the students learned about software artefact reviews as well continuous integration.
Code guidelines	EST, SQ	In the exams but also in exercises the students had to follow code guidelines and learned about the consequences if not followed.
Troubleshooting and Debugging	SQ, SE	The two topics were combined due to their correlation. Students learned to make out bugs and errors and how to solve them.
Versioncontrol	SQ, EST, SE	Git was an often-mentioned topic and was described in all three lectures. SVN was taught in detail only in SQ.
Testing	SQ, SE, EST	Testing was mentioned in all three courses as well. This included also Blackbox- and Whiteboxtests The students learned about the different kinds of testing concepts and practices like function testing or uni tests.

Table 4.2: Table of extracted concepts and practices

Listing 4.1 Defined entry questions for the interview

1. What did you do in the first steps of your work life after graduating?
2. How do you feel other people in your shoes experience their first steps?
3. Did anything of those steps change, the pandemic excluded?
4. How come that you decided to work after your bachelor's degree?

Listing 4.2 Defined questions for RQ1

1. How competitive did you feel compared to other people that studied at other universities or a different topic?
2. What did your job description require, that you learned at the University of Stuttgart and what was missing?
3. How did you expect your applying process would be till you have a job?
4. Which tasks were you able to handle faster because of what you learned at the University of Stuttgart?
5. Do you remember a challenge during the start you needed to deal with but didn't have the knowledge to handle it?
6. How did you feel you would be able to compete with other students from other unis in the field you wanted to work in?
7. How was the experience for people you know who also started into their work life after Bachelor?

4.3 Interview Questions

The questions were defined as described in 3.3. They can be found in the tables 4.1, 4.2, 4.3, 4.4 and 4.5.

4.4 Contacted students

The interviews were done over a time of two months. Table 4.3 shows how many students answered over which platform. The strategy to find contacts over friends was skipped because no graduate with a fitting profile could be found. In total 16 former students were contacted and seven interviewed.

A further detailed description of how graduates were contacted can be found in 3.3 Preparing Interview. Graduates who were interested in being interviewed answered within 24 hours. In the other cases, no answer was received. The time of the interview was initially set to 90 minutes but was changed to 60 minutes after no replies after the first 3 contacts on Xing. Hence, the time was

Listing 4.3 Defined questions for RQ2

1. What if you wouldn't have studied software engineering before starting your work life but still would have gotten your job?
2. What do you think were the most important topics in lectures and why?
3. What challenges in your work life so far did you overcome using knowledge of the uni?
4. With which knowledge was there a moment when you felt very lucky to have learned it?
5. ***VModel***: How do you see the relevance of the Vmodel in projects in which you participated so far?
6. ***Project management***: Which project and software development process models did you experience in projects and are you using them at the moment?
7. ***Requirements Engineering***: In the Requirements Engineering including analysis and specification processes that you experienced, how close was it compared to what you learned in the lectures and is it still relevant?
8. ***Architecture***: How is the architecture being planed compared to what you learned at uni?
9. ***UML***: Describe how UML appeared at the start of your job, and its relevance now?
10. ***Software Quality Assurance***: Did you perform reviews of software artefacts at work and is your company using static code analysis?
11. ***Code guidelines***: How different were the code guidelines you were taught in your job compared to the those at the University of Stuttgart?
12. ***Troubleshooting and Debugging***: How much of a base was the knowledge you gained about debugging and troubleshooting in your training and now?
13. ***Versioncontrol***: How much of a base was your knowledge about git and version control systems from the uni?
14. ***Testing***: Which kind of testing were you confronted within training and later?

Source	Contacted graduates	Replies	Interviewed
Using OPUS, finding graduates on Xing	6	3	3
Using Xing search engine terms Softwaretechnik and Universität Stuttgart	5	0	0
Using LinkedIn with fitting search terms	3	3	2
Using contacts of graduates who replied	1	1	1
Asking supervisor for making contact with former students	1	1	1

Table 4.3: Table of replied students

Listing 4.4 Defined questions for RQ3

1. Do you think that studying Softwaretechnik at the University of Stuttgart has improved the way you are working now?
2. Is there somebody at work that requires you to have knowledge about a topic you learned at uni?
3. Which work or project that you do right now still uses knowledge from University?
4. In which moment you were surprised that you can use a topic/concept that you learned at the university?
5. How is the experience for your friends in a similar situation?

Listing 4.5 Defined final questions to end the interview

1. Would you do it all again? Study SE at Uni of Stuttgart to start off your career?
2. Would you recommend others to study at uni of Stuttgart to start off their career?
3. What would you advise people to start their careers now?
4. What did you think of the questions you were asked?
5. Do you have any questions yourself?

set to 60 minutes. After this and the start of searching on LinkedIn more participants were found. The interviewed former students graduated in the years as seen in figure 4.1. Three of the participant graduated in the year 2019, two in 2018 and one each in 2017 and 2014.
Of all the participants, over the half worked as software developer right after their graduation as can be seen in 4.2. Two graduates became IT Consultant and one Software Engineer
In total seven graduates were interviewed.

4.5 Mind Map

With the used methodology, four Mind maps were created. Two for the research questions and two for the entrance and final questions. For the last two are passive Grounded Theory approach was used, since they were not the scope of this work. Therefore categories were still created and analysed but no explicit separation into sub, base, axial or selective categories was conducted. These Mind maps were then used for chapter 5. As said before after each interview the code was added to the Mind map and new categories created and connections made. An example for RQ1 of this can be seen in table 4.4. In the chapter 8 these mind maps are displayed enlarged for readability.

Figure 4.1: Graduation years of the interviewed graduates

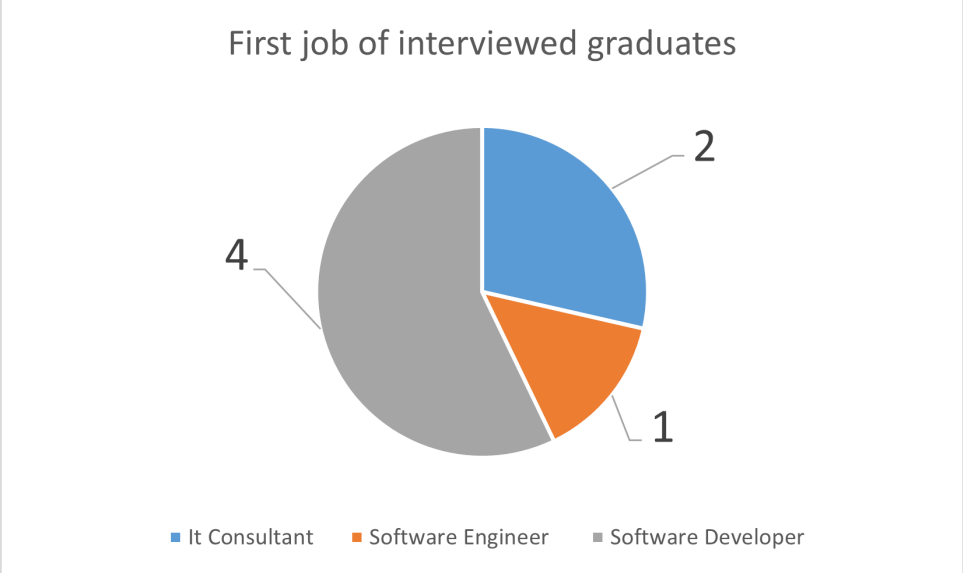


Figure 4.2: First job of interviewed graduates

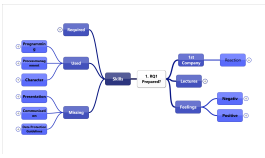
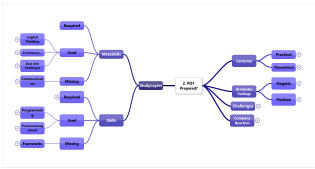
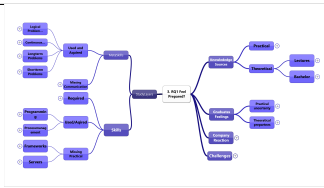
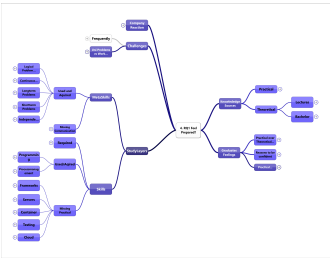
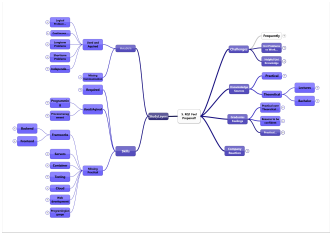
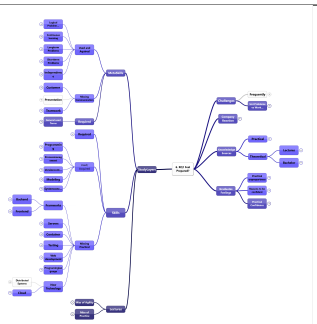
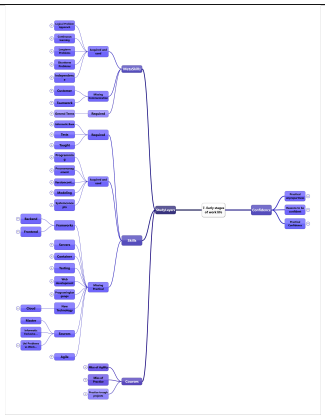
Frist interview	Second interview	Third interview
		
Fourth interview	**Fifth interview**	**Sixth interview**
		
seventh interview		
		

Table 4.4: Table of RQ1 Mind map evolution after each interview

4.6 Interviews

The questions were perceived well and gave graduates the room to answer freely. The interview time was changed to 60 minutes, but the number of codes extracted from the data did not decrease. Some questions were skipped in the interview, if they were already answered previously or in favour of more important questions like the concept and practice ones.

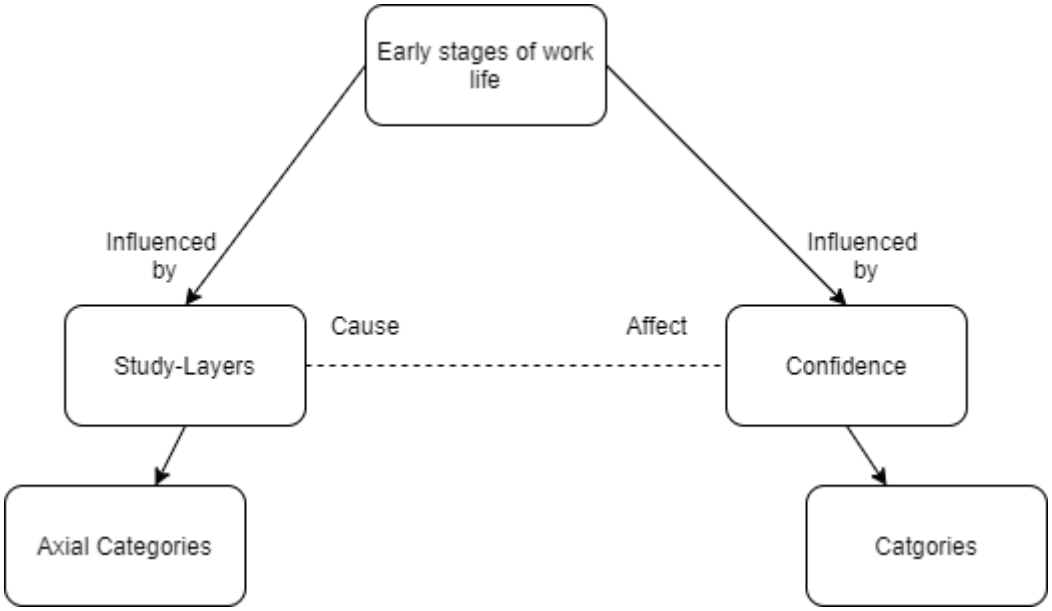


Figure 4.3: Relations between the top categories

4.7 Theoretical Saturation

As mentioned, after each interview the mind map got analysed and new categories noted. Theoretical saturation was reached after the seventh interview. The new code could be just sorted into already existing categories. After this, the Mind map was still revised as part of the final analysis. Even though new categories were created in that step, the content of those categories was from interviews before the seventh and served as an improvement of the display of the data.

4.8 RQ1 How well prepared did alumni of BSc-Softwaretechnik feel at early stages of their work life?

The mind map for this research question consists of 2 trees that are connected through one node. In the following sections for RQ1, RQ2 and RQ3, nodes, which represent categories, will be written like this [Category-Name]. The top node being [Early stages of work life] which connects the two categories [Study-Layers] and [Confidence]. But all three nodes have a relation to each other as seen in 4.3. The early stages of the graduates work life were influenced by their confidence and by their confidence and by the study-layers. The confidence was the effect of the study-layers influence on the graduates.

The categories [Early stages of work life] and [Study-Layers] were created as part of selective coding, while [Confidence] was part of the axial coding process. This is due to the Confidence-Tree only having one level of subcategories and therefore not justifying calling it a selective category. The same argument is used justifying [Study-Layers] being a selective category, due to its three levels of subcategories.
A picture of the mind map for RQ1, for displaying the depth, can be found here 4.4. Left is the

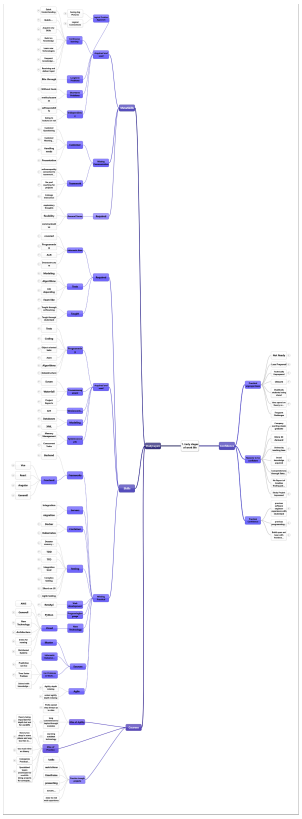


Figure 4.4: Mind map of RQ1

[Study-Layers] tree seen and on the right the [Confidence] tree part
A larger version of this Mind map can be also seen in figure 8.1 as well as a figure without the keywords as leaves in 8.12. In the following, the two trees are explained from top to bottom, gaining in details till the leafs. The leafs represent the statements of the graduates. The selective categories are explained in table 4.5.

In the following the three Study-Layers are further described followed by the Confidence tree. The three study layers build upon each other and are hence described as the first, second and third layer.

4.8.1 First layer: Courses

The axial category [Courses], being the first layer, is the source where the students learn skills. Courses describe here not just the lectures themself but also the gaining of information by taking up knowledge from it. May it be by going to the lecture or just examining the material created by the professor for the course. This also includes solving exercises provided by the course.

Selective Category	Explanation	Subcategories
Early Stages of work life	This category contains subcategories, which influenced the early stages of work life of the BSc-Softwaretechnik the in context of how prepared they felt. Due to the topics of this work, the subcategories are influenced by the training of the graduates at the University of Stuttgart.	[Study-Layers], 4.8.4 [Confidence]
Study-Layers	The Study-Layers are the developed theory for this research question. It describes three layers through which the students gain knowledge during study time. But they also display where knowledge was missing according to the graduates and challenged them after university	4.8.1 [Courses] 4.8.2 [Skills] 4.8.3 [Metaskills]

Table 4.5: RQ1 Table of selective categories

Theory over practice

The graduates criticized a high time investment in theory in the courses. They see the importance of theory for learning the concepts behind topics, but see it not being relevant for work life. This comes due to the companies preferring practical knowledge over theory knowledge, making it less worth. Specialised topics, which might be important for a scientific career then become irrelevant.

Practice through projects

The university offers two project courses, in which students have to develop a software in a set time. The Software-Praktikum and Studien project. The participants recognized the projects, especially the Studien projekt, for being close to a real work experience. They created user stories, worked through tasks, coded according to guidelines, participated in weekly scum meetings and presented the final work. They also needed to work with restrictions like a set time frame and using specific programming language or technology. In retrospect for the graduates, this came close to the work environment they were confronted within their early stages of their work life and now.

Missing agility

The graduates mentioned a lack of agility in the courses themself. A long commitment to topics through modules caused them to learn outdated technology. Mentioned here was Remote Method Invocation from Systemkonzepte und Programmierung. It is understood that the professors can not always stay up to date in the ever-evolving field of software engineering, but there should be a solution to that. One of them, that was proposed by a participant, is noted later in 6.5 Proposed improvements.

4.8.2 Second layer: Skills

Skills are the second layer. They are acquired through listening, processing and exercising during the courses. The subcategories describe which skills were required by their first jobs description, which skills they acquired in study and used at their job and finally which were missing and therefore challenging the graduate.

Required

This category is about what was required according to the job description. The answers in this category depended on the graduate, while three stated that they had learned everything required in university two stated they met the requirements thanks to their student job or self-teaching. The other two participants were required to have specific meta skills, which is listed later. The mentioned requirements were software engineering basics like: Databases, Programming, Algorithms and Data structures. During tests for their position two graduates needed to solve problems around modelling algorithms with a task that could reach a difficulty they experienced in exams.

Acquired and used

Students were able to use skills they picked up in programming like testing, coding, object-oriented languages, Json, Algorithms and Datastructures. Also in process management with Scrum, Waterfall and project reports. Versioncontrol with GIT, modelling with Databases and XML and Systemconcepts like Memory management and concurrent tasks. Which and how far, skills they learned helped them, will be further explained in the RQ2 and RQ3.

Missing practice

The skills missing as described by the graduates, were practical skills. Some of these skills were part of courses but lacking in depth for not applying them enough according to the participants. These are written like *this*. The graduates needed to acquire these themself or through their work. The following table 4.6 lists the missing skills and in which field they appear.
A missing topic that was mention from 6 out of 7 graduates was cloud technology. It was stated that they needed to learn it because of the ongoing trend in the industry.
As sources for these missing field were named, that they could appear in the master program, are included in a course which can not be done by Softwaretechnik students. Another source mentioned by 4 of seven graduates was the difference of tasks and problems given by the university and given by the job.

4.8.3 Third layer: Metaskills

Metaskills are the third layer and describe knowledge that was learned through learning skills. They represent a way of thinking, influencing the approach of tasks after the University. The subcategories show which metaskills the graduate required by job description, which they acquired and used and finally which were missing.

Field	Skill
Servers	Integration, Migration
Working with container	Docker, Kubernetes
Testing	Disaster recovery structures, *Test-driven Development* and *Test first development* as part of agile testing, Complex testing on security level, UI-Testing
API	REST-API
Programming languages	Python
Clouds	Basics for running a Cloud, Amazon Web Services, Cloud-Architecture
Agile development	Depth by exercising, scaled agility

Table 4.6: RQ1 Table of missing practical skills

Required

According to the job descriptions, the graduate or student needed to have flexibility, exploratory thoughts and be communicative.

Acquired and used

The graduates stated four different categories of metaskills, they acquired by studying Softwaretechnik at the University of Stuttgart. The first being a logical approach to problems with understanding logical connections and thinking abstract. The second is continuous learning. Five participants said they were able to quickly take new information in, understand it and use it. This way they were also able to acquire new skills like programming languages. Also pointed out was the ability to build upon acquired knowledge and refreshing it when needed.
The third was mentioned an acquired habit of prevailing to long term problems like a complex project. They also claimed to look for precision when it comes to solving short term problems like writing complex code for a user story. The fourth and final is independence. The graduates said, that because they needed to manage their schedule and studying for the courses themself they learned about self-responsibility. They could choose to visit a lecture or not and instead learn the provided material.

Missing communication

Missing metaskills were about communication. Participants talked about how they did not learn how to handle customers. That may be questioning customers for their needs in software or how to behave in a meeting with them. Two graduates mentioned a lack of learning on how to work in a team. This counts for working with other colleagues from the software field but also for colleagues that are not specialised in it. This was seen as a shortcoming because of software quality being connected to the teamwork ability.
Lastly, it was noted a lack of knowledge on how to do professional presentations for projects.

4.8.4 Confidence

This axial category shows through which feelings the confidence of the students were influenced. These subcategories are seen as a result of the required, acquired, used and missing skills as shown in Student-layers.

Practically unprepared

Graduates reported about frequently appearing challenges, when starting to work, and feeling practical unprepared for it. Two of them stated, that students, who went to a more practical university had an advantage, like students that finished a Dual study or whose university offered more projects. The learned theoretical base, which they all agreed on, is important but could have been also studied by themself they stated.

Practically prepared

The graduates who felt practically prepared for their early stages had previously gained practical knowledge with a student job, by programming in their free time or applying the learned theory in an open-source project.

Reasons for confidence

Even though some companies preferring practical knowledge over theory, it was stated that some companies also want the 'classic' student. That being students that have less practical experience so they can learn the guidelines of the company. Graduates with previous software project experience could be overqualified.
The graduates stated, that their Bachelor degree still helped them in their career by enabling options and reaching higher salary roles. Even though a lack of practical experience was noted, they also mentioned companies being impressed by the Study Project. An advantage that was mentioned was that they learned a very broad knowledge about software engineering and because it could choose from a broader field of jobs in the software industry. After all none of the graduates reported problems finding a job in Germany after studying at the University of Stuttgart.

4.9 RQ2 Which concepts and practices offered by the university have been applied in their work life

The mind maps for RQ2 and RQ3 were combined as mentioned before. The mind map splits again into two trees. In figure 4.5 is the mind map with all nodes, that are relevant for this questions, shown.
A larger version of this Mind map can be also seen in figure 8.7 in the appendix.
On the left is the tree of previously selected concepts and practices while the other side shows topics, which were mentioned by the graduates. This research question only focuses on the selected

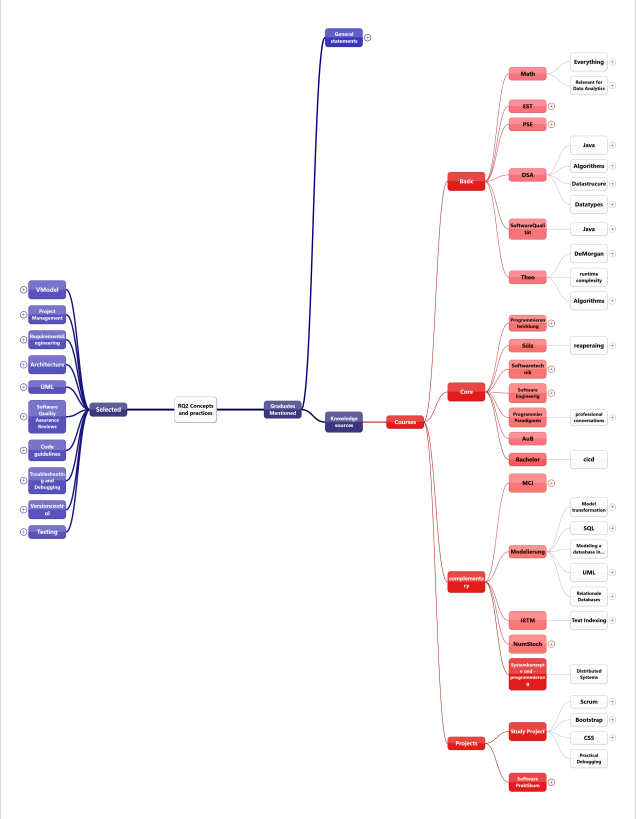


Figure 4.5: Mind map of RQ2

topics and the ones in the category [courses], displayed in red, on the right tree. The categories will be further explained in 4.10 RQ3, as we focus here only on naming concepts and practices which were applied.

4.9.1 Applied concepts and practices

The Category [Course] includes the subcategories [Basic], [Core] and [Complementary]. The classification was done according to the classification on the Campus system from the University of Stuttgart. [Basic] and [Core] lectures are courses which have to be completed by the Softwaretechnik students, while in [Complementary] are courses which students could choose from four catalogues. From these catalogues, they had to choose five modules. Even though the question only includes the concepts and practices, the courses were included as well. It turned out that graduates could not remember all of the names of the concepts they use during work. This was due to the names being less important during working the graduates stated. They rather connected them with courses and hence they were included as well.
The [basic] concepts and practices can be found in table 4.7, the [core] in table 4.8 and the [complementary] in table 4.9.

Courses	Practice or Concept
Mathematik für Informatiker und Softwaretechniker	
Programmierung und Software-Entwicklungr	Object oriented programming, Java, Datastructures,
Theoretische Grundlagen der Informatik	DeMorgan, Algorithms
Datenstrukturen und Algorithmen	Java coding, Algorithms, Datastructures, Datatypes
Software-Qualität	Code guidelines, Troubleshooting, Debugging, Versioncontrol, Testing
Einführung in die Softwaretechnik	VModel, Project management, Requirements engineering, Architecture, Software quality assurance, Code guidelines, Versioncontrol, Testing

Table 4.7: RQ2 Table of basic concepts and practices mentioned by graduates

Courses	Practice or Concept
Algorithmen und Berechenbarkeit	Runtime complexity, Algorithms
Programmentwicklung	
Sichere und zuverlässige Softwaresysteme	
Software Engineering	VModel, Project management, Requirements engineering, Architecture, UML, Troubleshooting, Debugging, Versioncontrol, Testing
Software-Praktikum	
Programmierparadigmen	
Studienprojekt	Scrum, Bootstrap, CSS, Debugging
Bachelor	

Table 4.8: RQ2 Table of core concepts and practices mentioned by graduates

Courses	Practice or Concept
Modellierung	Model transformation, Relational databases, SQL, UML, XML
Mensch Computer Interaktion	Eight golden rules
Systemkonzepte und -programmierung	Distributed Systems, Concurrent Tasks
Einführung in die Numerik und Stochastik für Softwaretechniker	
Information Retrieval and Text Mining	Text indexing

Table 4.9: RQ2 Table of complementary concepts and practices mentioned by graduates

As can be seen in the tables, the previously selected concepts and practices of Softwarequalität, Einstieg in die Softwaretechnik and Software Engineering were included as well to have a complete list of which were applied. A more detailed view of those topics can be seen in 4.10.
Mathematik für Informatiker und Softwaretechniker, Programmentwicklung, Sichere und zuverlässige Softwaresysteme and Programmierparadigmen did not have any concepts, for none specific ones were named.
In this listing were also included the programming projects Studienprojekt and Software-Praktikum because the students learned new concepts and practices as well and applied them. Though the concepts and practices they learned are individual as the projects have different requirements every semester, like the topic of each Bachelor thesis.

4.10 RQ3 To which extent concepts and practices as offered by the university were found to be helpful in their work life?

The Mind map displayed in 4.6 is split into two trees. In this figure, leafs are hidden to be able to show the complete Mind map. Therefore only the selective-, axial-, and base-categories are shown. The left tree shows the previous selected concepts and practices, while the right side refers to concepts and practices mentioned by the graduates.
A larger version of this Mind map can be also seen in the appendix in figure 8.3.

In the following, the axial-categories for the selective-category [Selected] are listed, after this the axial-categories for the selective-category [Graduates Mentioned].

4.10.1 VModel

Overall, the VModel had a mixed feedback.

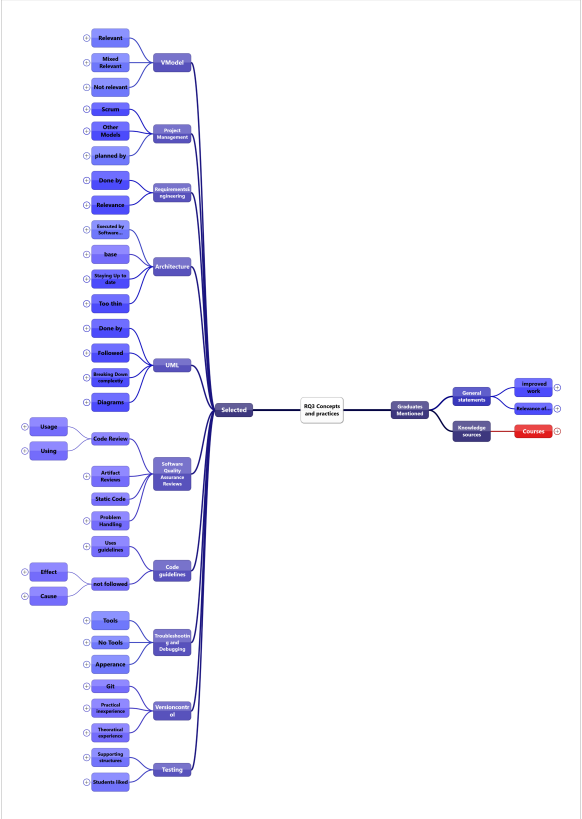


Figure 4.6: Mind map of RQ3

Relevance

Graduates stated the Vmodel being important for still being used in project planning and being an important part in understanding the company structure as it is widespread.

Mixed Relevance

It was also stated as being relevant but could also be ignored, due to Scrum being a more important model.

No relevance

Two alumni stated to have not worked with the VModel and not seeing use in it.

Selective Category	Explanation	Subcategories
Concepts and Practices	The main topic of the whole Mind map is the concepts and practices, which the graduates learned when studying.	[Selected] [Graduates mentioned]
Selected	The category contains concepts and practices, which were selected before the interviews from the chosen courses. The graduates were specifically asked about these in the interview to find out to which extent they were helpful.	4.10.1 [Vmodel] 4.10.2 [Project Management] 4.10.3 [Requirements Engineering] 4.10.4 [Architecture] 4.10.5 [UML] 4.10.6 [Software Quality Assurance] 4.10.7 [Code guidelines] 4.10.10 [Troubleshooting and Debugging] 4.10.11 [Versioncontrol] 4.10.12 [Testing]
Graduates mentioned	The node contains categories that are not based on the selected practice and concepts. Students were not asked directly for the experience with specific topics. Therefore, the content for the concept and practices were mentioned or brought up by the graduates during questioning.	4.10.13 [General statements] 4.10.14 [Knowledge sources]

Table 4.10: RQ3 Table of selective categories

4.10.2 Project Management

Scrum

All the participants used Scrum as part of agile development. Due to scrum being defined for smaller teams, alternative versions of it are used in companies, also to adjust to developing software for a fixed price. Scrum was stated to be an important base for the working of the industry and for understanding the company structure. It was argued that the theory itself about scrum was not enough to understand and use later, but the Studienprojekt helped, as it showed its practical value. They also stated that using scrum practically in university in projects, was close to projects in the industry. Through this, they were able to understand the scrum process better in the company.

Other models

After Scrum, Kanban was mentioned the most as being a lighter Scrum. Other models used were SAFe and the waterfall model. Last only came to use before Scrum was introduced. Even though companies are not officially using the waterfall model anymore, few graduates argued that their company is still using it mixed with Scrum. This comes from the management team working given tasks like in a waterfall model, but the development team using the agile manifesto.

4.10.3 Requirements Engineering

Executed by

It was stated, that the Requirements engineering was done by a Product Owner or Requirements Engineer. Therefore, the consultants and developers were not directly involved in it.

Relevance

Graduates learned the fundamental basics, even though they pointed out a difference to what they learned in university. They used their knowledge to read documents from the Product Owner or Requirements Engineer. They said they also needed it, to work on User Stories, as they did not always state all the requirements. Students that did not use Requirements Engineering yet saw value in it for later in their career.

4.10.4 Architecture

Important for base knowledge

Two graduates stated that in their company the software architecture was worked on by the Software architect. But even so they did not work with it themself, the architecture taught in university is seen as an important practice for their work lives. Graduates used their knowledge to understand connections, system landscapes and system structures. The opinions on design patterns from university were mixed. While three graduates stated to never use the patterns, the other four saw the patterns as important for software quality and wished for more programmers using them.

Theory focus

Criticized was the focus on theory when it comes to learning about software architecture. The students wished for more practice as they also felt like not understanding the design patterns when starting to work and needed to catch upon knowledge.

Staying up to date

The students also saw it as important to be up-to-date with current software architectures as the keep value for saving costs and time when applying them.

4.10.5 UML

Followed

Two graduates saw UML as part of the Requirements Engineer or Product Owner as they provide most of them to the developers. Besides one participant, UML is used by all the alumni. UML is not executed as detailed as in university, which is only needed when code is generated from a diagram. The practice was emphasized for being relevant, useful and reappearing.

Breaking down complexity

The Graduates used their UML skills for breaking down complexity in their work life. They used it as a communication tool for presenting software parts to customers, colleagues or management, who may be or may be not, from the software engineering field. Alumni also used UML to organize in a development team and to read information from the product owner.
The graduates see it as an everyday tool for model reading, big data display, working on interfaces and documentation.

Diagrams

While UML overall was the topic of this question, the most stated diagrams were class diagram, sequence diagram and activity diagram.

4.10.6 Software Quality Assurance

Code Review

All interviewed graduates stated to participate in code reviews frequently. Merge- and Pull-Requests are reviewed by other developers and have to run through a pipeline as part of continuous integration. This is done to counter possible development mistakes as early as possible to save time and money. The graduates argued, that the circle of reading user stories, coding, testing and reviewing was not taught practically enough in university. They felt a gap in using a pipeline for pull requests as they did not use one in university. Graduates said that even though they learned it in the Studienprojekt by themself, they would have proffered to get more practical experience beforehand in these areas.

Artefact Reviews

Artefact Reviews with reviewing the specification, like done in “Einstieg in die Softwaretechnik“ were reported in only one company. The graduate stated it never working properly because roles were assigned but no one was following them.

Static Code Analysis

The static code analysis tools Sonarcube, as well as Lint, were used by five of the 7 graduates. They were familiar with the concept but needed to learn to use them in their early stages of work life.

Problem handling

It was reported that found problems through code review, artefacts review or static code analysis were not always solved due to the weighting of costs. The problems were then ignored because of the set time frame for the software development and therefore set money.

4.10.7 Code guidelines

4.10.8 Use of guidelines

The code guidelines were stated to be different in each company, though being not as strict as in university. The code guideline taught was seen as a good base and are called still relevant. They see code guidelines as important to be able to write customer-friendly, colleague friendly and high-quality code. Code documentation, speaking identifiers and design patterns were named.

4.10.9 No followed guidelines

The graduates stated to be regretful if code guidelines like taught in the university were not used at their company. They commented that not following guidelines has the effect like learned from the university. They reported about complex methods, rare commenting or oversized classes. One reason for this was named the weighting of costs by the management which leads to code being quickly produced with less quality assurance.

4.10.10 Troubleshooting and Debugging

Tools

The only tool that was mentioned by a graduate to have been used in university for debugging was FindBugs. He stated to not have used it again after that time.

No tools

The graduates stated to have not learned about tools for debugging in university. Another alternative to writing print statements could not be named. Nevertheless, it was also reported that graduates saw this as an advantage as they were able to understand code better by breaking it up and testing it logically than colleagues that were used to tools. They also argued that this was useful when code could not be debugged with a tool due to its complexity. One example named was concurrent threads.

Appearance

Even though the alumni learned about the topic in university, they felt like there was theory missing. As well as troubleshooting and how to handle it.

4.10.11 Versioncontrol

Git

The graduates stated that only Git was used for no better alternative existing. It was stated before companies moved to Git they were using SVN.

Theoretical experience

The graduates stated to have understood the concepts in university and had enough time for it, with the courses paying enough attention to it. They understood that Git would be important for their jobs.

Practical inexperience

On the other hand, they stated that they still were missing practical experience in Git when they started into their work-life. Two graduates also stated they felt lost in the beginning as they needed to catch upon knowledge. They had problems in handling complex commands like breaking up pull requests. Some topics they felt lacking in were Git Workflows, Pipelines and DevOps.

4.10.12 Testing

Test appearances

Four of the seven participants had worked with Whitebox and Blackbox tests, while it was also mentioned that the names play less of a role in their work lives. They were able to test functions for extremes and limits. They also were able to handle unit tests.

Advantages of Knowledge

The graduates saw themself having acquired broad knowledge over testing concepts in software engineering.

Knowledge gaps

The graduates felt missing a practical experience in UI testing and Test driven development, included in agile testing. As mentioned before, the graduates felt a gap in using pipelines and connected involved tests. Mentioned tests were End-To-End tests and smoke tests. A general lack of knowledge in integration tests was also noted.

4.10.13 General statements

This category includes collected statements, which concern all concepts and practices, which the graduates learned at the University of Stuttgart.

Improved work

Six of the seven participants agreed that the university improved their working. They stated that because of those concepts and practices they were able to solve complex issues with a logical approach, think abstract and analytically. They also reported to be more meticulous when it comes to the quality of software and following code guidelines. Another advantage pointed out was, that they were able to have professional talks with colleagues. But it was also stated, that this base of theoretical content of concepts and practices, the graduates could have also learned themself.
One graduate stated that the university did not improve his working and more improvement of skills were needed.

Relevance of names

Throughout the interview, graduates could not always remember the specific names of concepts or practices. One of the reasons for this was that some names were less relevant in work life. This follows from practices just being used because they are part of a routine, without being named. They are processes that need to be handled to move on without much importance of the name of the step. Another reason is the fluent knowledge of the university till now. The graduates could sometimes not tell if they learned about a practice in university or during a student job, as their university time was at least two years ago.

4.10.14 Knowledge sources

This category included sources of concepts and practices which were mentioned by the alumni.

Courses

This category covers helpful concepts and practices in courses. Further explained in 4.9 RQ2.

5 Additional findings

The goal of the following questions was to get additional information besides the research questions. Therefore questions could be also closed questions. For the answers, the Mind maps for the entrance and for the final questions were used. The Mind map for the entrance questions can be seen in figure 5.1 and for the final questions in figure 5.2. An enlarged version of the entrance Mind map can be seen in figure 8.4 and of the final questions in figure 8.5.

In the following text, studying is interpreted as to study Softwaretechnik at the University of Stuttgart.

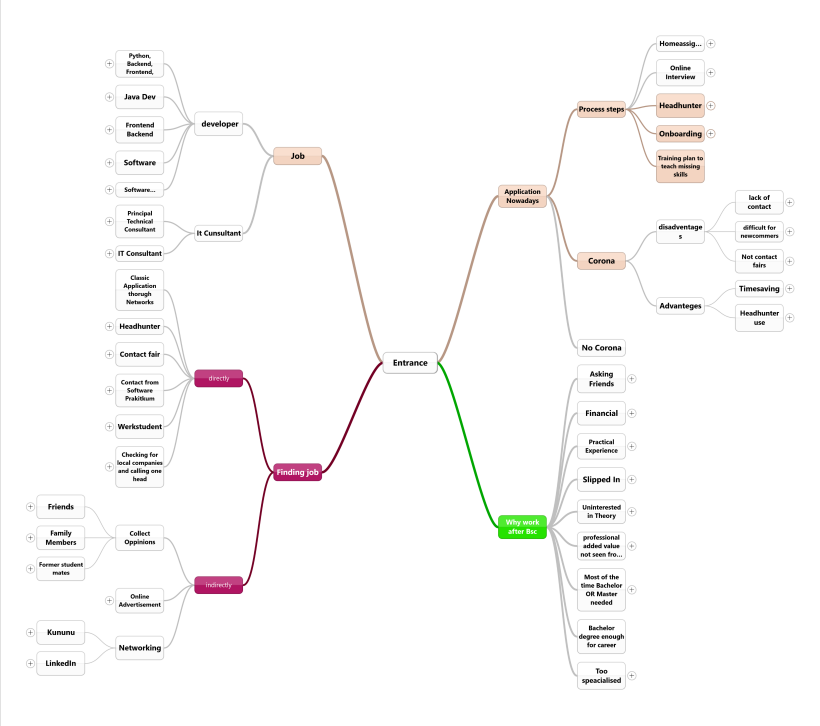


Figure 5.1: Mind map of the answers for the entrance questions

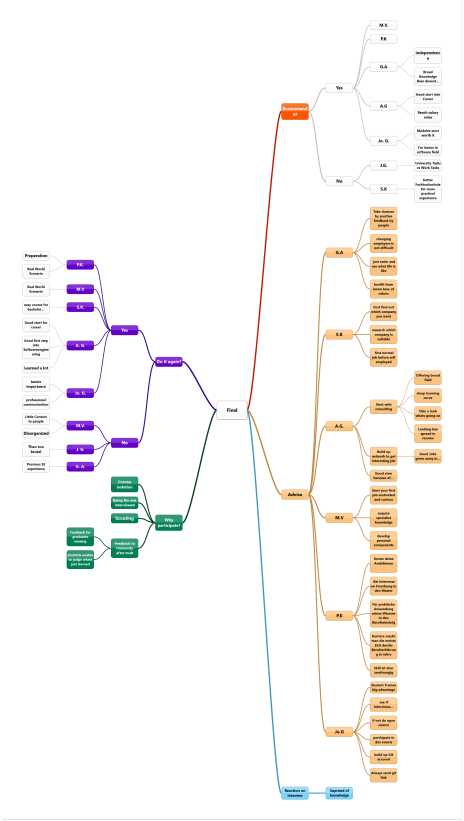


Figure 5.2: Mind map of the answers for the final questions

5.1 How did the graduates acquire their jobs?

Four of the graduates continued working in the same company they had a student job in, after university. They found the student jobs through a student contact fair, by making contact with a company during the Software-Praktikum and by contacting their company directly after research. The other three graduates came to their job similarly. By applying after research, through a student contact fair and by using a headhunter.

5.2 How would the pandemic have affected the graduates in searching for a job

At the time of this work, the ongoing pandemic is affecting graduates due to the contact restrictions. Because of this, contact fairs are cancelled, and the application process needed to change with interviews being online as well as the onboarding. The questioned graduates saw this as a time saver due to being able to do the application process from home. They argued they would have searched an online platform like Kununu, LinkedIn or Xing and paid attention to the advertisement for home office jobs. They did not see a restriction in their networking activities, which is a good source for finding a work possibility argued by them. The participants also saw a possibility in using a headhunter, with paying attention to the motivation.

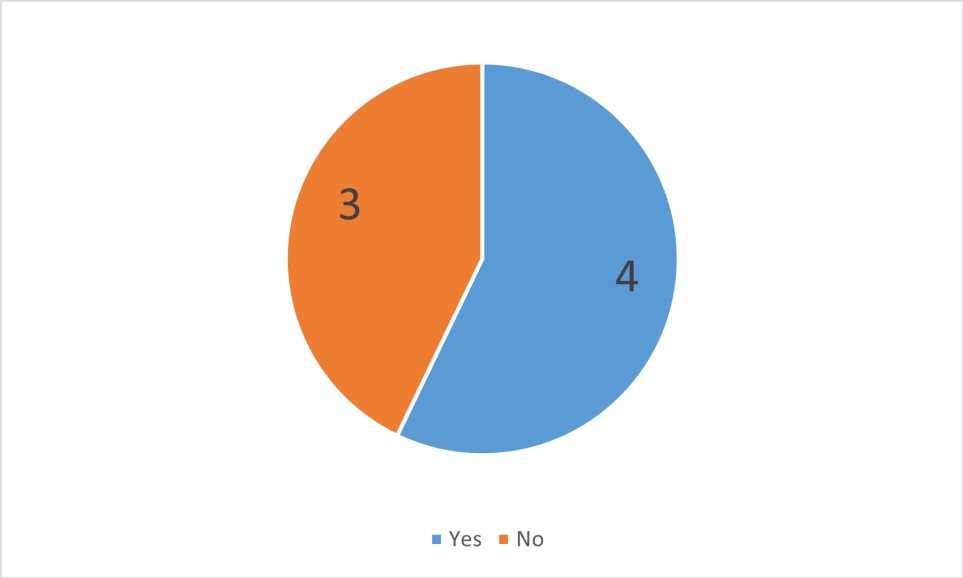


Figure 5.3: Would the graduates study at the Uni Stuttgart Softwaretechnik again to start their career?

5.3 What made the graduates decide to participate in the Interview?

As mentioned before the participant were all eager to share their experience after university. Some of them were in a position of normally interviewing others, so they saw this as an opportunity to be the being asked questions.
The participant also saw this as an opportunity to give feedback to the university as they did not receive a chance after university to do so. This enabled them also to give ideas for improvements. Another reason is the ongoing pandemic. The graduate stated that the interview was a change from everyday life working from home.

5.4 Would the graduates study at Uni Stuttgart Softwaretechnik again to start their career?

As you can see in figure 5.3 four of seven participants would start their career again studying Softwaretechnik at the University of Stuttgart. The following states their arguments. The outcome of this question also depended on the personal situation of the graduate and also depends on previous software development experience. Even though it is written here as the argument of the participants, each one of them had a different opinion.

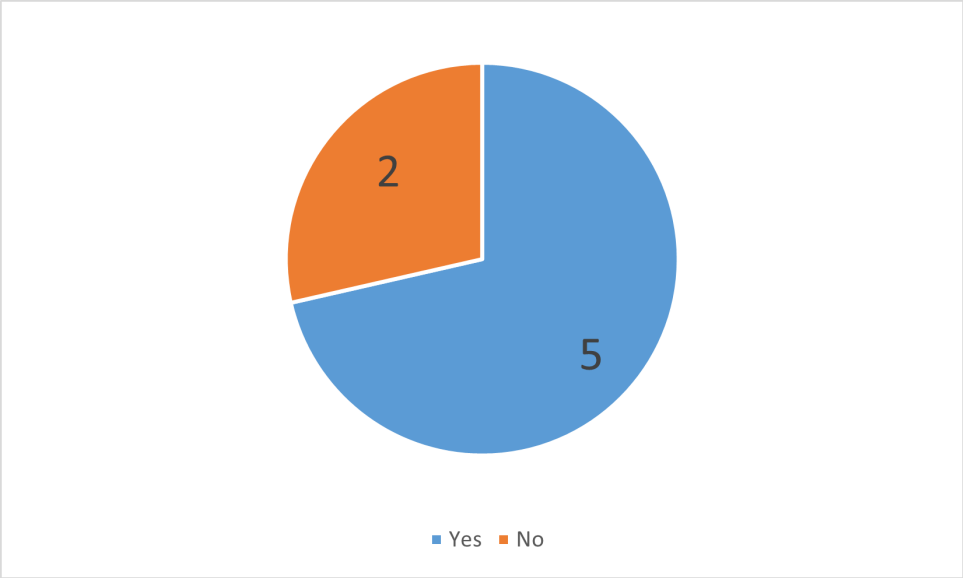


Figure 5.4: Would the graduates recommend others to study Softwaretechnik at the University of Stuttgart?

5.4.1 No

The graduates argued that the University was too unorganized for them, and it was hard making contact with people. Also mentioned were classes which were too time-intensive for not being relevant for the future for them. Another argument was that previous software engineer knowledge made studying at the University of Stuttgart obsolete.

5.4.2 Yes

The participant who would study again stated that the study provided a preparation with real-world scenarios for their work life. It was also said that the easy courses enabled for an easy bachelor degree. For them, it was a good start for their career and a good first step into their working career. They learned about the importance of basics in Software Engineering and felt enabled to have professional communication.

5.5 Would the graduates recommend others to study Softwaretechnik at Uni Stuttgart?

As can be seen in figure 5.4, most of the graduates recommend others to study Softwaretechnik at the University of Stuttgart. Others in this scenario means people who are thinking to study in the field of software engineering. Compared to the previous section it shows that some graduates would not study again but recommend others in doing so.

Listing 5.1 How come that the former student decided to start their work life?

- Their previous acquired student job enabled them to directly start their job
- Their financial situation required them to start working
- They saw too much specialisation and unneeded additional theory in doing the Master
- Graduates saw no added value in doing master and after research stated that companies were looking for Bachelor and Master graduates treating them as equals.

5.5.1 No

The graduates saw more value in studying at a more practical oriented university and stated that the university tasks in exercises were not relevant for their work life even though working in the software engineering field.

5.5.2 Yes

Graduates that previously had said to not study again, still stated that they learned to be independent and learned a broad knowledge which did not bind them to a specific field in software engineering. Other graduates that would study again and recommend it, said they had a good start into their career and the degree enables them to reach higher salary roles. Another argument was, that the modules, which prepared them for their work life.

5.6 How come that the former student decided to start their work life?

For the graduates, there were multiple reasons to start into their work life which can be seen in list 5.1.

5.7 What advice can the alumni of BSc-Softwaretechnik give to now graduating students?

The graduates are at a point of life where, with master or not, students who study will likely end up in. Therefore this was a good opportunity to ask these people what they would recommend and advise students, who want to start into their work life now or soon. The results were summarized in the following sections.

5.7.1 Take chances

Students should collect feedback from friends and contacts about their experience of the company they work in. If a company is recommended due to positive feedback or they are invited personally they can just dive in and gain first experiences. This way they can benefit from the know-how of others. If they do not like their first job they can still change employers since it is not difficult. Once the students find the job they want, they gain work experience over the years.

5.7.2 Do Research

The student should do their research about what kind of company they want to work in as well as which company would be suitable for them. They should also research what their ambitions are and what they are aiming for.

5.7.3 Consider advantages of IT Consulting

It was recommended for students to start with IT Consulting because of it offering a broad field. Even though Consultants deal with different technologies in each project, the resume would not show it. Otherwise, changing the field multiple times can be seen perceived as undetermined.

5.7.4 Networking

It is recommended to students to be active in communities in the field they want to work in and exchange with people there. Building up a network increases their chance of getting interesting job offers as many jobs are given away internally in company and contact networks.

5.7.5 Stay curious

Students should start their first job curious and motivated. During their job, they should acquire specialised knowledge and develop personal components. This way they gain also value.

5.7.6 Build up a Git Account

The graduates advise building up a Git Account. Work on projects that students upload and add the link to their resumes. The projects should involve frameworks, that the companies they want to apply for are using. This shows the students practical experience, also if they did not have a student job while studying.

5.7.7 Do not give up

If students are having trouble getting a job they should increase their networking but also try themself on open source projects and upload them to their Git Account. Programming events give them the perfect opportunity to work on new projects, gain practical knowledge and build up their network.

5.7.8 Just start

The most important point, that was mentioned by most of the participants, is to just start into work life, to take opportunities and gain experience. Students should just see how it works out as their degree enables them for good job opportunities and thanks to digitisation many companies are searching for workers in the software engineering field right now.

6 Discussion

6.1 Reflection of the research questions

6.1.1 RQ1 How well prepared did alumni of BSc-Softwaretechnik feel at early stages of their work life?

The results indicate that the graduates did not feel prepared after their University time. The confident graduates acquired it through practical experience from a working student job or their hobby. This study reported missing skills in practice and communication through courses being not agile and a focus on theory over practice. Still, it was shown that the graduates gained a set of metaskills allowing them to learn the needed skills in the early stage of their work life. Also, even though a lack of practice was reported, the projects were mentioned positively and see as a good source for acquiring practical skills. Acquiring broad knowledge was also mentioned, as an upside of the university.
A source of practice, which was not mentioned was the exercises and homework of the courses. This may be due to the exercises seen as part of the theory by the graduates or the tasks not resembling problems as they appear in their work life. Another possibility is that the participants could not remember the exercises they did.

Interpretations

This means, that even though the graduates spend at least three years at the university in Software Engineering, they felt like they did not acquire the skills to be confident through the university alone. Some of them worked in a student job to counter the theory focus. On the other hand, none of the graduates reported having problems finding a job in the Software Engineering field in Germany. For the graduates the time after university is a notable moment, therefore it would have been mentioned because it is an influencing factor in the early stages of work life. Also, even though a lack of practice is reported by them, three would go to university again and five recommend it to others to study, as shown in section 5. This shows that they still see it as a good education for starting a career in software engineering.

Implications

The graduates reported on how they felt prepared in the early stages of their work life, but from a perspective that allowed them to also know if they were right to do so. At the time of the interview, they had gained, at least two years of, experience in the industry. This way they could compare the

knowledge they gained since university with their knowledge from university when they started to work. The results can help the University of Stuttgart as feedback as well as the Institute of Software Engineering. In section 6.5 possible Improvements for both are listed.

Limitations

The results do not proof, that every graduate is missing or acquired these skills. They represent a summary of what was reported in the interview. Every graduate had a personal experience of the university time as they had different connections to their fellow students to study with and courses offering different topics. The two projects, as well as the bachelor, can include different topics as well, which can support the graduates in their early work life. Nevertheless, all interviews show a common pattern of the graduates favouring the practice over theory. The practice included the one being done in projects and the one missing. The reported fields in which the university is lacking according to graduates can still be used to improve in favour of coming student generations. This can also help the participant and the industry when more confident students come into the labour market, as companies will need to spend less time and money on onboarding programs. What is also not considered is possible causes besides the university that influenced the felt preparedness of the graduates. These could also be dependable on their work environment.

Recommendations

This is why further research should be done into which factors influence the confidence of alumni in their early stages of work life. Another topic to further research should be how universities, teaching software Engineering, can stay agile in the quickly changing and evolving software industry to offer new trends like cloud technology.

6.1.2 RQ2 Which concepts and practices as offered by the university have been applied in their work life

In section 4.9 the concepts and practices mentioned by the graduates were listed.

Interpretations

The noted concept and practices allocated to their courses show almost complete coverage of the offered courses for Softwaretechnik students. The table for basic courses represents exactly the basic courses Softwaretechnik students need to absolve in their first two semesters according to the Campus system. Therefore containing knowledge, which is helpful and relevant for the work life of the students. Almost the same goes for the core courses. The only modules not included in the list are “Einführung in die Technische Informatik“ and the case study “Fachstudie“. Also, here is shown that the courses contain valuable information. The complementary courses can not contain all the offered modules in this category, but “Modellierung“ is standing out as it has the most mentioned concepts in that category. Even though this module could be skipped by a student, it has important concepts for software engineer students like SQL to handle databases. It as well

contains the practice of UML, which is also a part of the mandatory software engineer course software engineering. Therefore it should be considered to move this module to the core courses for the Softwaretechnik study.

Implications

The results show that the graduates were able to use these concepts and practices in their work lives. It gives the University and the organizers of the courses feedback that the listed content is remembered by the graduates and is still applied. Even though Versioncontrol and Testing was dealt with in three courses, the graduates still felt to be lacking in knowledge about it after university, as shown in the discussion of RQ3. While redundancy is also good for recalling the knowledge of students, this result could be used to reduce redundancy in courses.
Overall it shows that graduates can apply the content of the software engineer specific courses and it helps them in their work.

Limitations

Just like in the discussion for RQ1, tables in section 4.9 are a summary and do not give information on which job required which concepts and practices. Also, the absence of the concept, practice or courses does not indicate that it was not helpful as the study is based on the interviews. They have the disadvantage that the participant only has the set time frame to remember used concepts and practices. This is why the course names were included as well, as the graduates seemed to remember them better.
Depending on which profession the graduates came to do after university, they required different concepts and practices. Therefore the graduates were also asked for their contacts from their study time, to find out which concepts and practices they were using. But it turned out that the participants had mostly lost contact with their fellow students or they had gone on, to do the Master in Softwaretechnik. That is why only three professions were covered: Software Developer, Software Engineer and It Consultant. But even so theoretical saturation was reached after seven interviews as the professions shared common used concepts, practices and opinions. Still, almost all of the basic and core software engineer courses were mentioned.
Another limitation was, that the more experience the graduate had, the bigger was the chance that he could not tell apart knowledge from and after university. But this could not be changed as more experience meant also more knowledge about how helpful the concepts and practices were.

Recommendations

Further research should be done into which job in software engineering required which concepts and practices offered by the University of Stuttgart. It is recommended to interview master alumni for this, who started to work, as they represent a bigger population than bachelor alumni who started to work. This way it can be found out which jobs can be covered with the broad concepts and practice offer.

6.1.3 RQ3 To which extent concepts and practices as offered by the university were found to be helpful in their work life?

All selected concepts and practices were useful to the graduates. Troubleshooting and debugging was reported to be useful, but the graduates felt like they were lacking in theory and tools. FindBugs was not used by them after university. In Software Quality Assurance, the artefact review was done in only one case and not successfully. The reviewing of code and the connected development cycle was felt missing in practice also for static code analysis they would want more knowledge. The VModel had mixed feedback for being helpful by graduates, as some used it to understand the company structure, not used it but seeing relevance in it and not used it. For versioncontrol, the graduates felt that they acquired a theory base, but were lacking in practice, also connected to the development cycle. The Research Engineering they learned, in university, helps them for working on user stories and is therefore always helpful as it is a part of their development routine. What they learned about architecture, the alumni used for understanding connection in software. The design patterns received mixed feedback, but the graduates who use it see it as important knowledge as they apply it to improve their software quality. But it was also agreed on a lack of practice. Every company has their own code guidelines, but the university taught once are seen as a helpful base and desirable for code regulation. They use it to improve their code in readability and complexity. With one exception, is UML seen as an everyday tool with frequent use for breaking down the complexity of software components. This makes it one of the most helpful topics for the graduates. The alumni learned Scrum in project management and everyone has applied it in their company. They learned about the theory in the courses and understood it with the practice in projects and could use it since the early stages of their work life.
Considering all concepts and practices, the graduates had also reported feeling like these skills improved their work as they acquired metaskills through them, like a logical approach to problems and abstract thinking.

Interpretations

The extend of how helpful a practice or concept is, is based on if the alumni could apply it and if they had the required knowledge for it. To be able to use a concept or practice they learned, the alumni needed to have understood it in theory and practice. This is the balance of theory and practice, as mentioned in the introduction. As theory is important to understand the base of a skill and what it does, practice is important to know how to use it, also in connection to other steps in development. Concepts and practices like UML or code guidelines can be practised with exercises and while this can be also done for Scrum, requirements engineering, architecture, software quality assurance, troubleshooting and debugging, version control and testing, these should be practised more as part of a development cycle. In a project, like the Studienprojekt the students can then apply them as they would in their work life.

Implications

The result can be used to see which concepts and practices are still requiring more theory, like debugging and scrum, or practice, like code reviews, troubleshooting and debugging, Git, testing and architecture. It also shows which have enough theory like Waterfall model, architecture, testing,

Git, or practice like Scrum and UML. Course content could be changed based on this, to balance the theory and practice of topics. The results also show, that not just the learning of these concepts and practices is important for the work life of the students but also the metaskill, which they learn. As almost all graduates agreed to work better due to those metaskills. These can be not directly taught but influence the graduates working, as they can use them to learn new concepts and practices and handle problems.

Limitations

The interviewed alumni graduated at least two years ago, which leaves the possibility that criticised aspects were already changed. Therefore does this work not represent a state of the balance of theory and practice for current students. But the result can still be used to compare it to the courses, concepts and practices now. One of the other limits of this listing is, that depending on the profession of graduates some concepts or practices can be of more help. As no requirement engineers were interviewed, the requirements engineering comes only in use for user stories. As mentioned in the last discussion, asking the graduates for the experience of their fellow students did not solve this problem.

Recommendations

Due to the previous limitation, is recommended to interview master alumni which have at least two years of experience. They serve as a broader population and therefore, possibly more variety in professions. With two years in the industry, they also have gained more knowledge and left the onboarding program. With research and therefore an active approach of the alumni there is a bigger chance of getting feedback since the alumni would have direct contact.
It should be also considered to add more practice into the previously mentioned concepts and practices through projects.

6.2 Results compared to related work

What distinguishes these results from the studies listed in chapter 2 Related Work is, that the research questions are asking for the skills, like in RQ1 and for detailed information about the use of concepts and practices like in RQ2 and RQ3. Also, the interviewed population were graduates of the University of Stuttgart. These graduates were then asked questions about previously selected content they learned at the University in software engineering courses, which was also not done in the other works. Therefore, this work was able to get feedback for the organizers of the software engineer courses as well, as it focused on the specific learned concepts and practices from those courses.
The results are similar to the results of previously done studies. In [CJNZ20] a major trend for agile software development in software engineering education was reported. The participants had argued that multiple agile methods were lacking in theory and practice like Test driven development and Scrum, which supports this trend from the side of the graduates as well.
A similarity between this work and [RW13] is that a lack in communication skills is reported. This included talking to customers or colleagues, which is referred to, in the related work, as oral

communication. It also notes a lack in testing and teamwork. But while in the related work a lack of project management was reported, the participant in this study told that they were able to apply scrum successfully, due to the practice of the Studienprojekt.
[MMCF05] had noted a lack of exposure to teamwork, management and practical experience by former chemical engineering students. This work did not report on acquired management skills, but independence was noted due to needed self-responsibility in university. Like in the related work, this study also reported on graduates lacking practical experience for their workplace in concepts and practices.

6.3 Validity threats

In the following sections, the most relevant in qualitative review according to [Max92].

6.3.1 Descriptive Validity

Descriptive threads in this study come from the process of writing said answers in the interviews. Therefore the interviews were completely recorded. The answers of the graduate were then written down word by word down in a spreadsheet. To not modify answers in translation from German to English, a Google translator was used. The resulting text was then checked for resembling the answer of the graduate. In the same spreadsheet, the text was then turned into code. This process can be revised and hence this thread is under control.

6.3.2 Theoretical Validity

The process of turning interview answers into code held a risk of code not representing what was actually said and context getting lost. Therefore, code was compared to the answer again before placed in the Mind map to make sure it still resembles the answer and the context is paid attention to. Another thread was, that the context of a code could be lost over time. A skill could end up in a different category due to a software bug of the Mind map or an unnoticed manipulation. Hence, Mind maps were saved after each interview individually and then compared to the previous version to ensure the context or category of code did not change unnoticed.

6.3.3 Generalizability

As stated in [SF20] “generalizability is limited by the sample that is studied“. Therefore is external generalizability not guaranteed. Also, because master graduates have additional knowledge with different or new concepts and theories. This counts also for graduates of other fields or from different universities. This study was chosen to be a sample study to be able to reach maximum internal generalizability, over a specific population, in this case, bachelor Softwaretechnik alumni of the University of Stuttgart, which started to work afterwards. Internal generalizability is controlled because alumni from four different graduation years were interviewed till theoretical saturation was reached. This means no new theory would be found within this population. Therefore, the result can be generalized for the bachelor Softwaretechnik graduates.

6.3.4 Interpretive Validity

The interpretive threads can be seen in drawn conclusions from the created mind map. As this study was worked on by one researcher, Interpretative Validity can not be guaranteed. To reduce this thread as good as possible, the Mind map and the drawn results were compared on different days to have possible different mindsets. The results were also compared to previous related work.

6.3.5 Evaluative Validity

No evaluation as described by [Max92] were made in this study as all claims are drawn from the data.

6.4 Challenges

6.4.1 Handling large amounts of data

Recorded answers for the interview questions needed to be written in text and into code, to be placed into categories. For noting the answers, Microsoft Excel was used as it allowed to create rows for each stage of the text transformation. To manage the codes by placing them in categories and creating new ones, a Mind map was chosen. This allowed an overview of the codes and made connections visual. If the Mind map reached a size to be too big to be displayed, nodes could hide their leaves.

6.4.2 Keeping categorisation in mind when editing Mind map

One problem with the Mind map was, that it was easy. A distinction between the kinds of categories of Grounded Theory could be then disregarded. Therefore, before each analysis, the new categories were put in context and coloured depending on a sub-, normal-, axial-, or selective-category. A node with a darker colour described a more abstract category than a lighter one.

6.4.3 Considering students who started with a student job in work life

Student job was considered as a part of their early work life, as the participant continued working at those companies after university. The decision was made due to the student job also being a first encounter with the industry and therefore a part of the early stages.

6.4.4 Contacting graduates before knowing if theoretical saturation is reached

To fulfil theoretical saturation, the interviews needed to be analysed separately after each one. It was also to be expected, that contacted graduates would not answer immediately. Therefore, four graduates were always contacted at once. On reply, the interview dates were then placed in a way, so an analysis could be done and then being followed by the next interview. So interviews needed to be planned before clearance if theoretical saturation was reached. This challenge became obsolete as fewer contacted graduates answered after the fourth interview, therefore giving enough time after the analysis to decide if more interviews are necessary.

6.5 Proposed Improvements

6.5.1 For University

Getting study specific feedback

The feedback system of the University of Stuttgart for graduates was unknown to the participant and the before mentioned planned feedback system for IT fields, failed due to lack of participation. It is therefore crucial to spread awareness of these systems, as the feedback of graduates with work experience is important for the development of the university and the content the students learn. The graduates can resemble feedback from students but also from the industry.
The university could ask the graduating student if they want to be notified in two years, to participate in a questionnaire. The emails could be then sent automatically.
Another possibility is to have studies like this one done in each study field, by a bachelor student for their final theses. This way, they also have to review content they learned and get inside knowledge into where they might be in a few years. By collecting advice from the graduates, current students can also benefit from this. Hence, university and students get feedback. The university can balance their theory and practice and adjust their content, and the student learns which knowledge they still might be missing.

6.5.2 For institute of Software Engineering

Use of mini-courses

One challenge which software engineering modules face is to bring the students up-to-date with trends. As described before, it can be difficult to teach new appearing technologies with modules that are taught for one semester. One idea would be to use mini-courses to teach new topics and be agile. The mini-courses would be taught in a month and could change their topics when necessary and be agile. One of the participant had the idea to use online courses for this, and cooperate with a platform offering them. This way the professors could concentrate on their courses and would not need to handle more content. These mini-courses could be also applied to have projects where the students use their learned knowledge in a work-like environment.

Projects for depth of knowledge

The mini-courses can be projects to let students learn the agile development cycle before the Studienprojekt. These projects would be like the Studienprojekt but shorter and needless coding. This way they can concentrate on teaching how concepts and practices the students learned work together when developing software. In this project, using scrum, the students could plan the software with UML, set up a project on Git, create a work environment to make builds of the software, set up a pipeline for CI/CD, define code guidelines and integrating them with static code analysis tools, write tests before coding, debug code, and test the finished software. This could be done by having a group of three to five students go through these steps together, or create roles like architect or project manager. The last would make it closer to work-life as the project manager plans the project, the architect designs the structure, and the developer receives user stories from the requirements engineer.
One of the skills can be focused to have the students learn in-depth about it. These small projects could be done also as part of an exercise, but then could only concentrate on it for a limited amount of time during the week. One idea would be to have a one-week project where the students can then only concentrate on it without having to handle other lectures in that time. It could then also replace an exam.

Professional coaching for teamwork in projects

Teamwork and communication were lacking metaskills mentioned before. A graduate had the idea of professional coaches join the projects. They could be invited from cooperating companies and would have the goal to improve the teams's effectiveness through communication. This would help the human to human interaction between the students and potentially improve the software quality [LSD+16], but also teaches about teamwork as it is a part of software development.

7 Conclusion

Universities have to balance practice and theory to prepare students, so they feel confident in their early stages of work life. Confidence is the effect of the study layer's influence on the students. The study-layers are courses, skills, and metaskills. Courses teach skills and through learning those skills the graduates learn metaskills. Courses were reported, by the graduates, to be too much theory, but the two projects were seen as important preparation for work life. Criticized was the missing agility of the courses, not being able to pick up on trends in the industry. The alumni saw themself having acquired base skills in software engineering, but said to also missing practical skills in different technologies, which were relevant for their early stages. They acquired a set of meta skills to be able to learn new skills efficiently and handle shortterm-, longterm as well as logical problems. Reported missing were metaskills for communication important for teamwork and customers handling. But the graduates had also stated there were reasons for confidence as they had acquired a broad knowledge and the study helped to start their career.
From the previously selected concepts and practices, VModel, Scrum, requirements engineering, architecture, UML, software quality assurance, code guidelines, troubleshooting and debugging, versioncontrol, and testing, all were seen as helpful and the graduates could apply them in their work life. Though it was also reported that the graduates felt missing practice in code reviews, troubleshooting and debugging, version control, testing, and architecture.
The graduates had expressed to not feel practically prepared but saw the advantages of the bachelor degree for their career. They reported having acquired a broad knowledge and named a wide range of concepts and practices being helpful.
The lack of felt preparedness comes from missing skills, but also from missing practice. The graduates needed to have a student job to counter that and to have a confident start into their work life after achieving bachelor. The two projects in their Softwaretechnk bachelor study were mentioned positively by the graduates and show that it is a great source for practical knowledge as some concepts and practices were understood better afterwards. To include more projects like them but shorter the study time could make the students understand the theory better they learned. It could also improve their teamwork, let them have more work life-like experiences, feeling more prepared and confident after their bachelor.
The balance of the theory and practice is an important task for the universities as well as the organizers of the content of the courses. To do so requires feedback from former students who gained experience with their acquired knowledge in the industry. Of the interviewed participants, none knew of the feedback system by the University of Stuttgart. It is therefore advised to make students as well as graduates aware of this or have frequent studies for each field, in favour of current and coming students.

8 Appendix

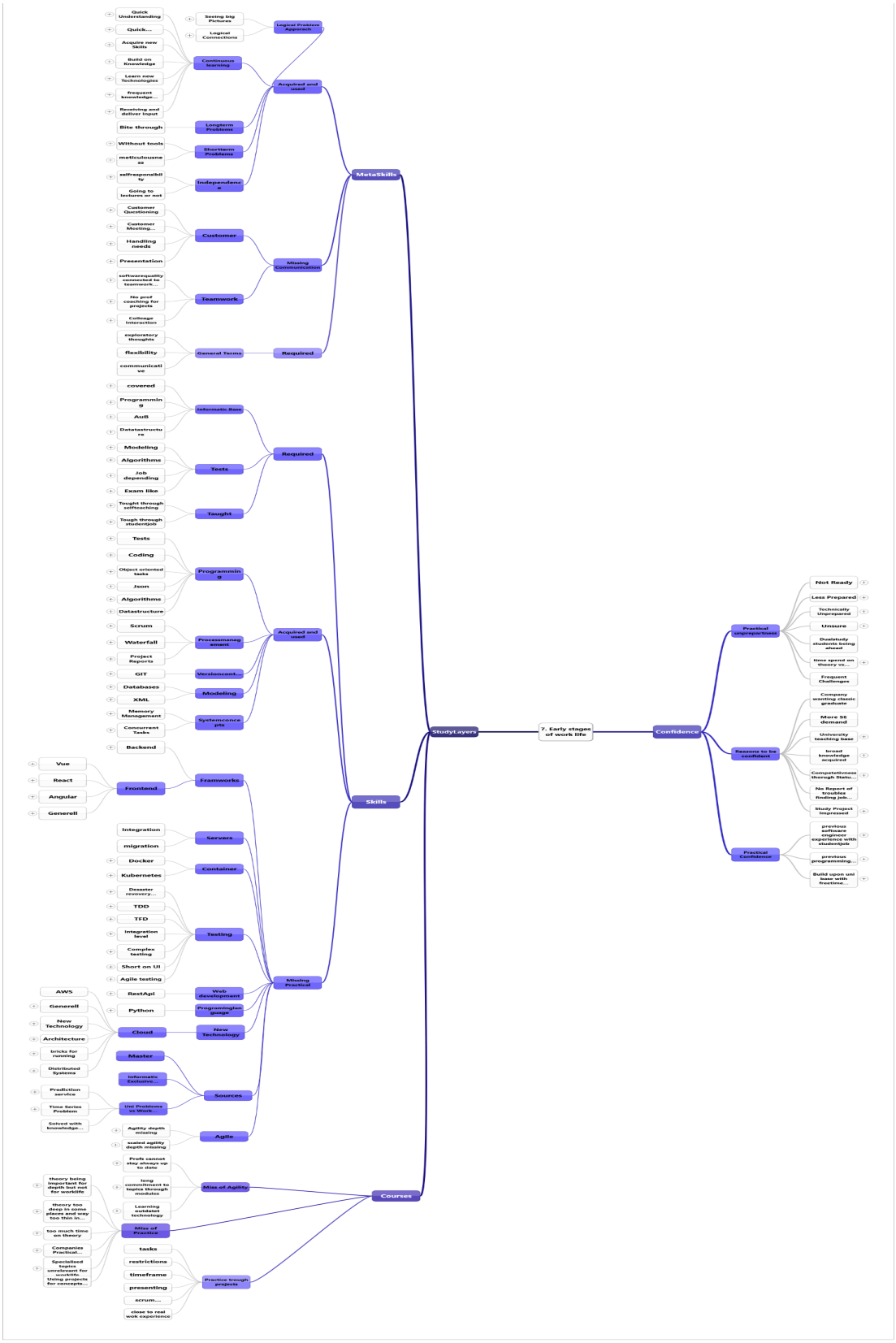


Figure 8.1: Mind map of RQ1 on a full page

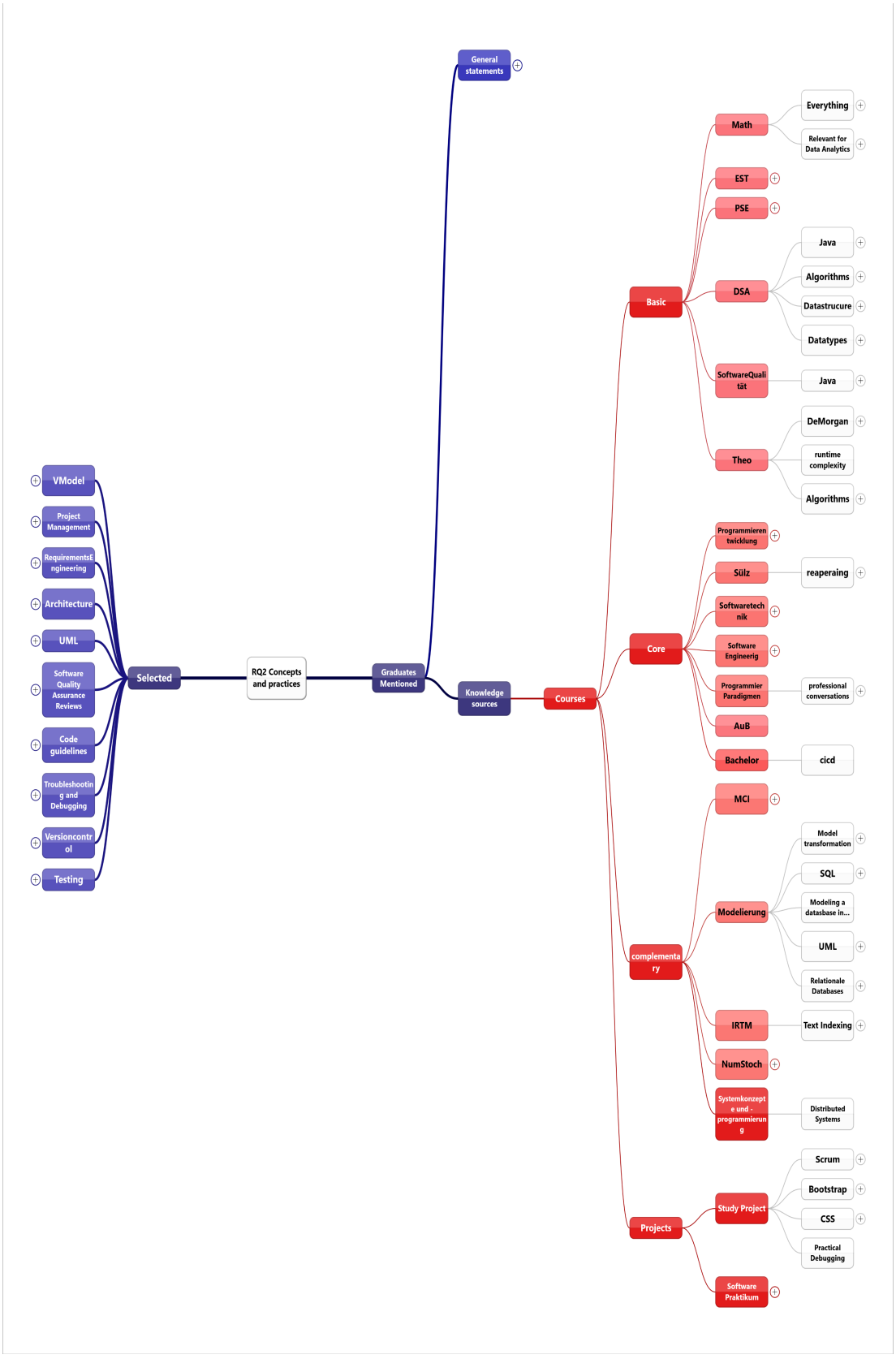


Figure 8.2: Mind map of RQ2 on a full page

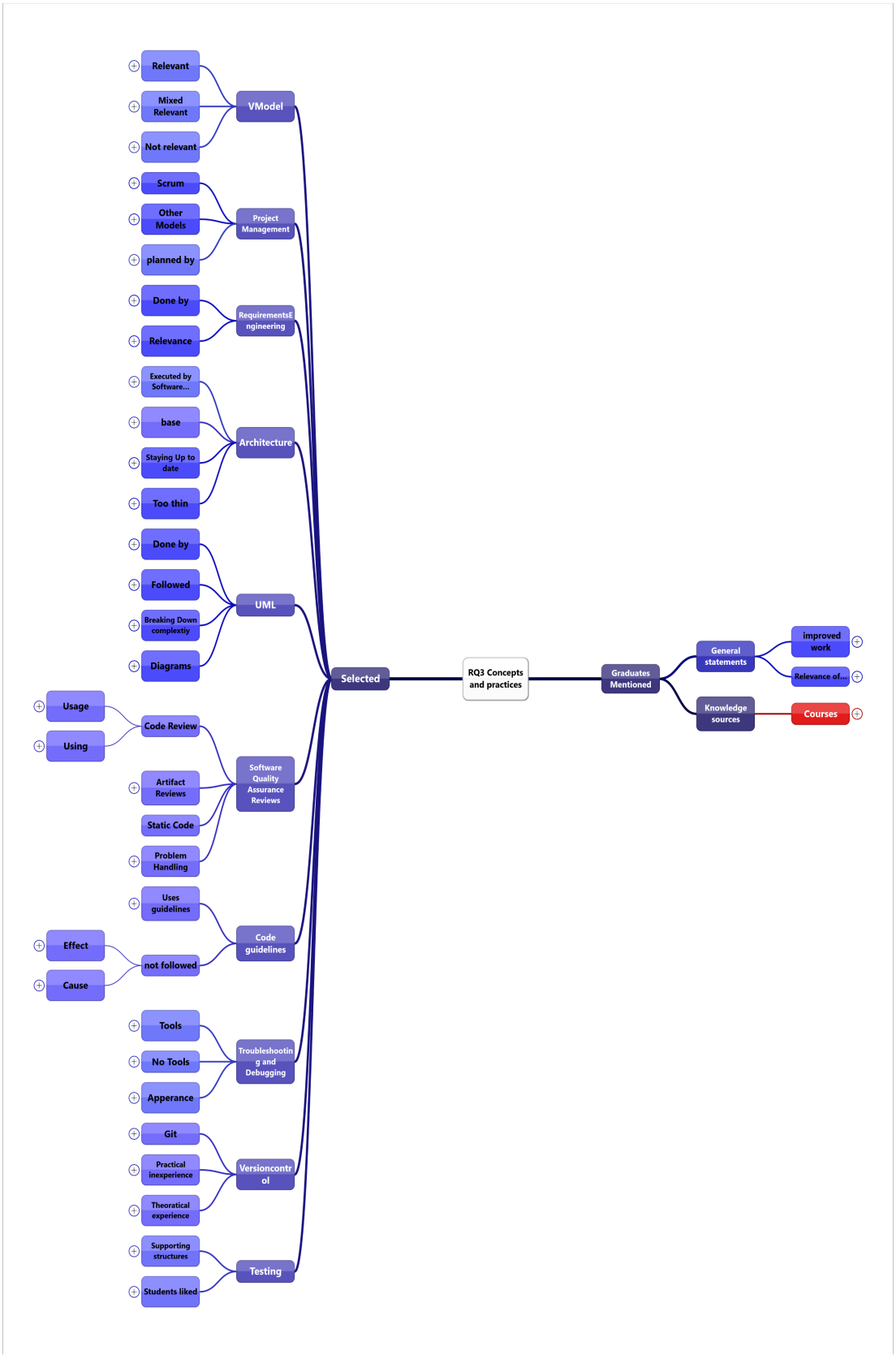


Figure 8.3: Mind map of RQ3 on a full page

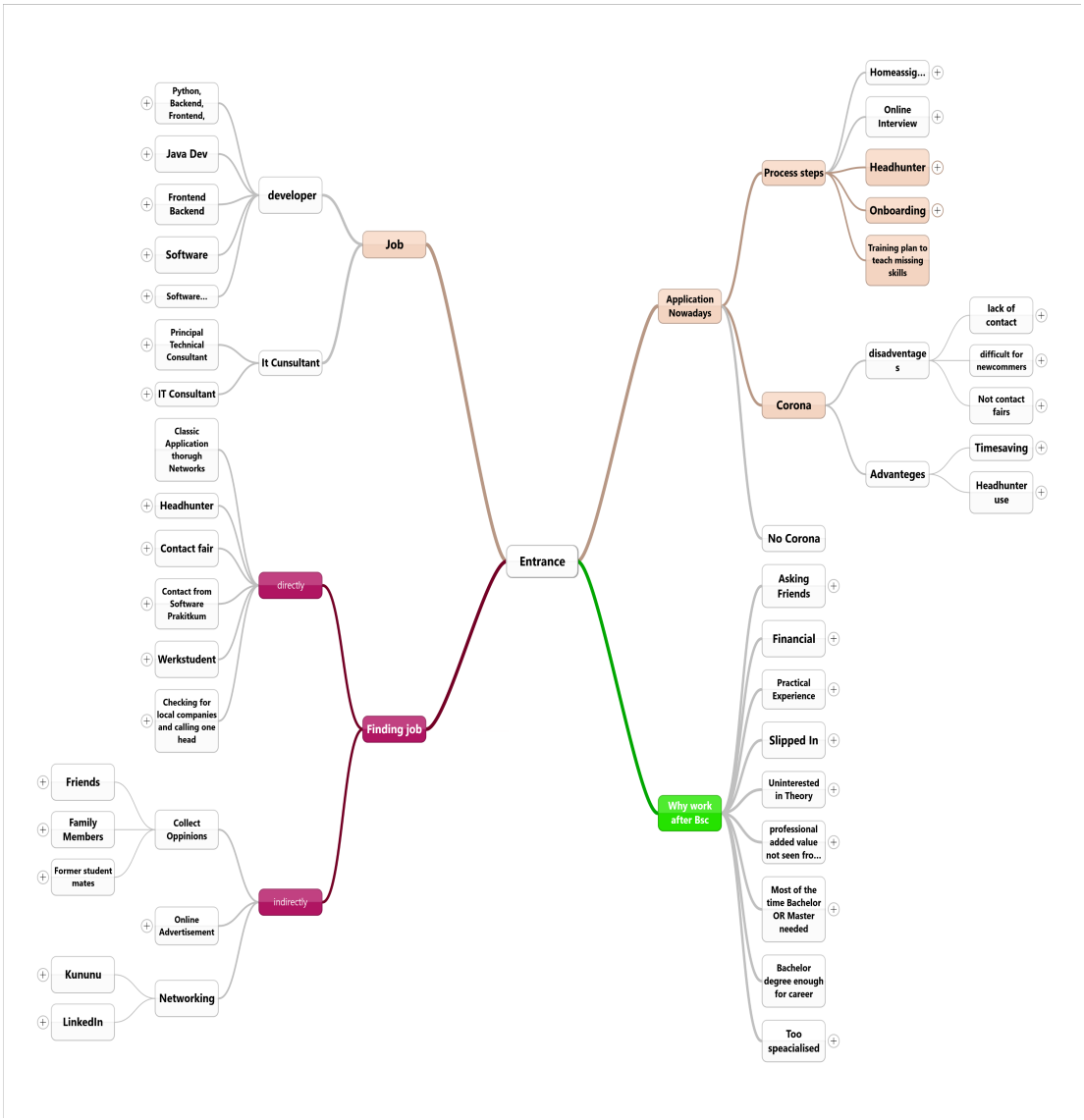


Figure 8.4: Mind map of the entrance questions on a full page

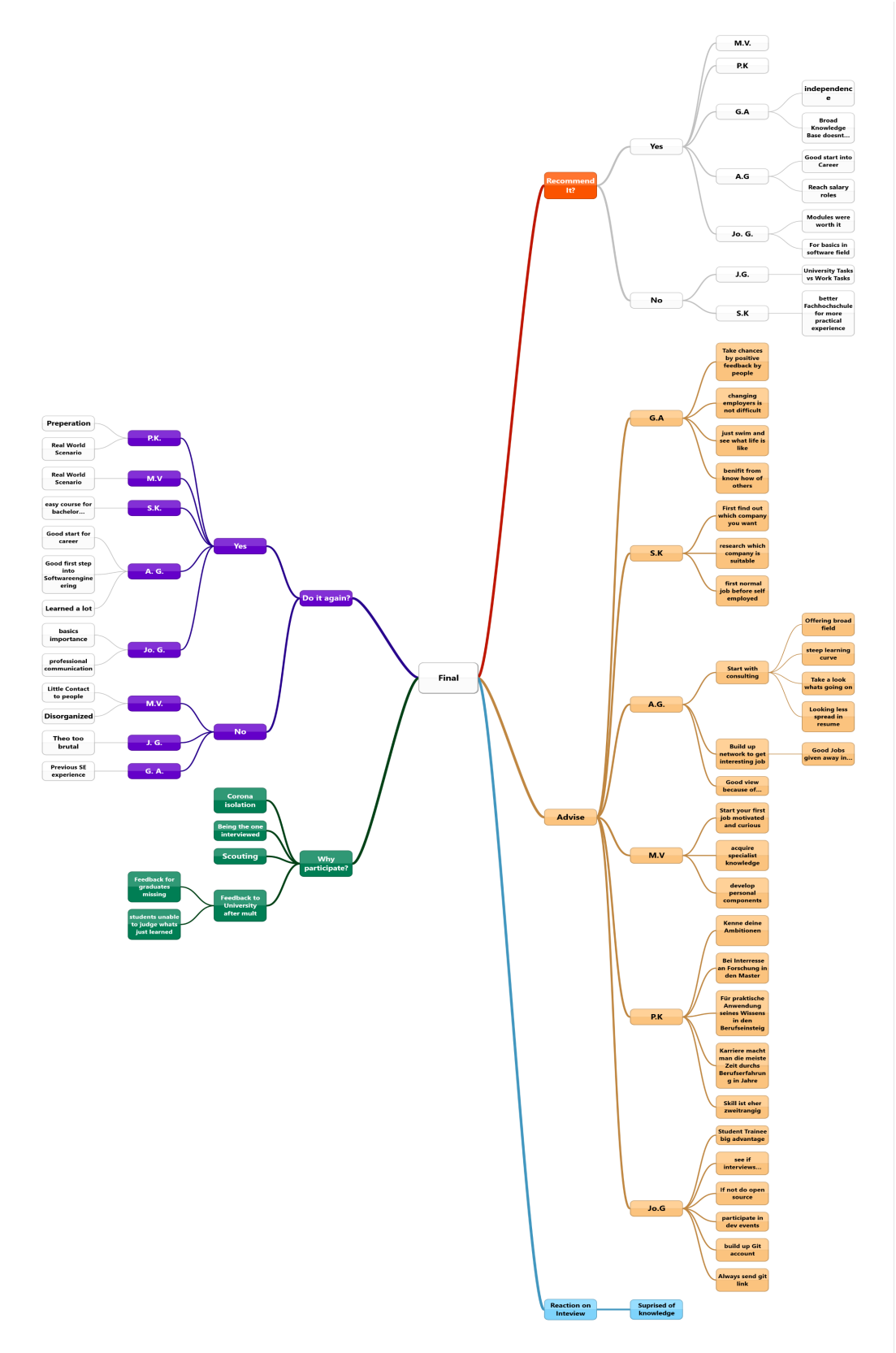


Figure 8.5: Mind map of final questions on a full page

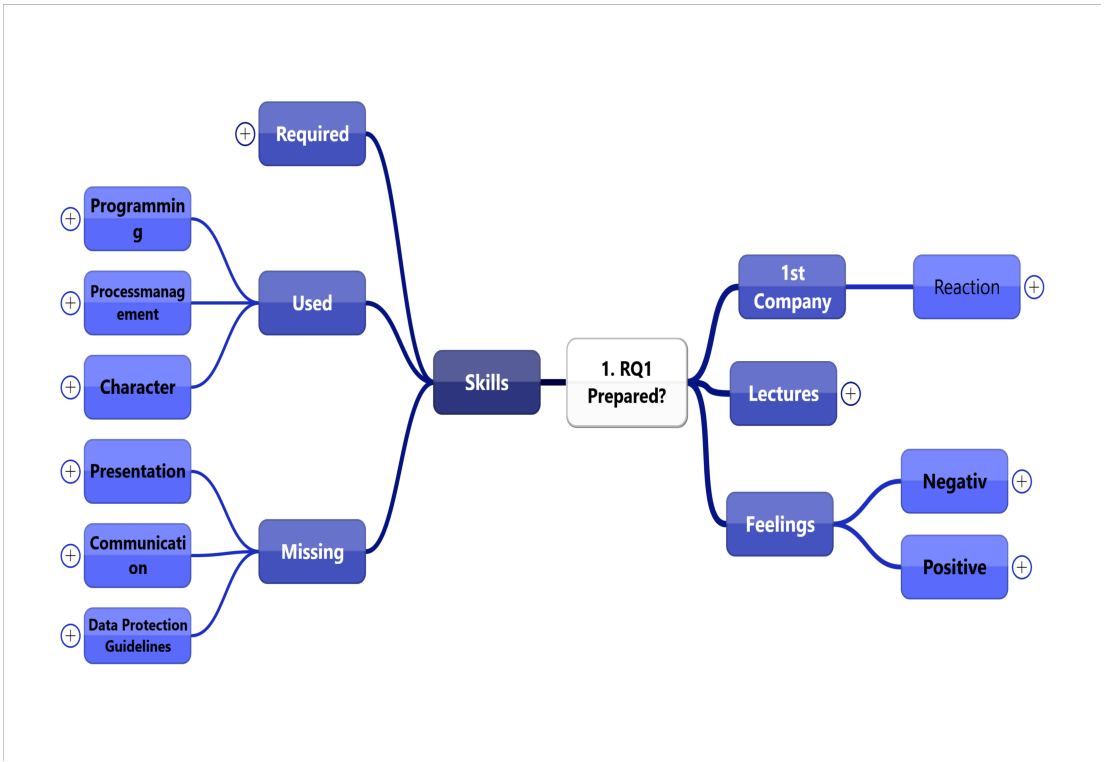


Figure 8.6: Mind map of RQ1 after the first Interview

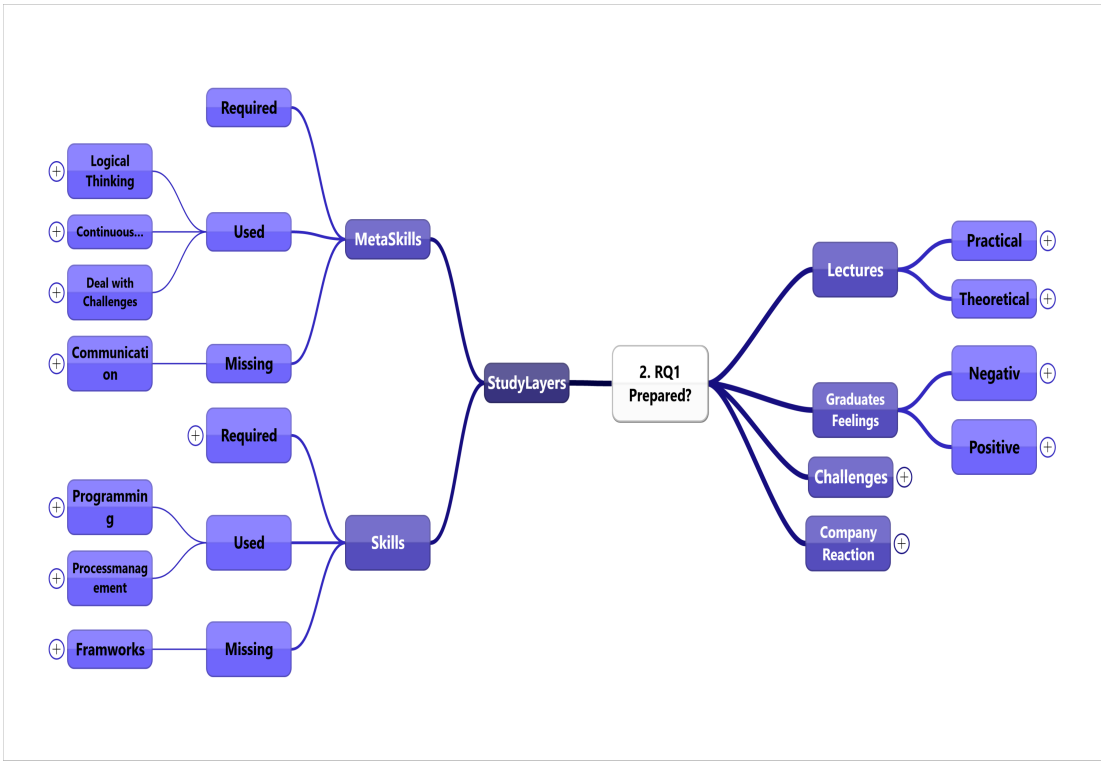


Figure 8.7: Mind map of RQ1 after the second Interview

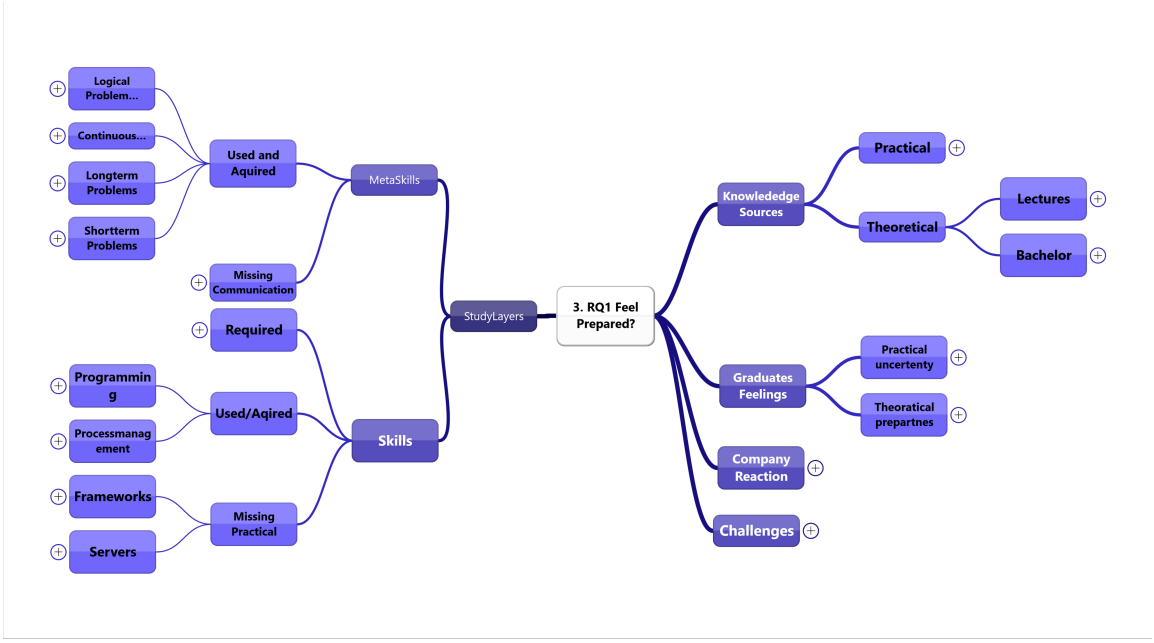


Figure 8.8: Mind map of RQ1 after the third Interview

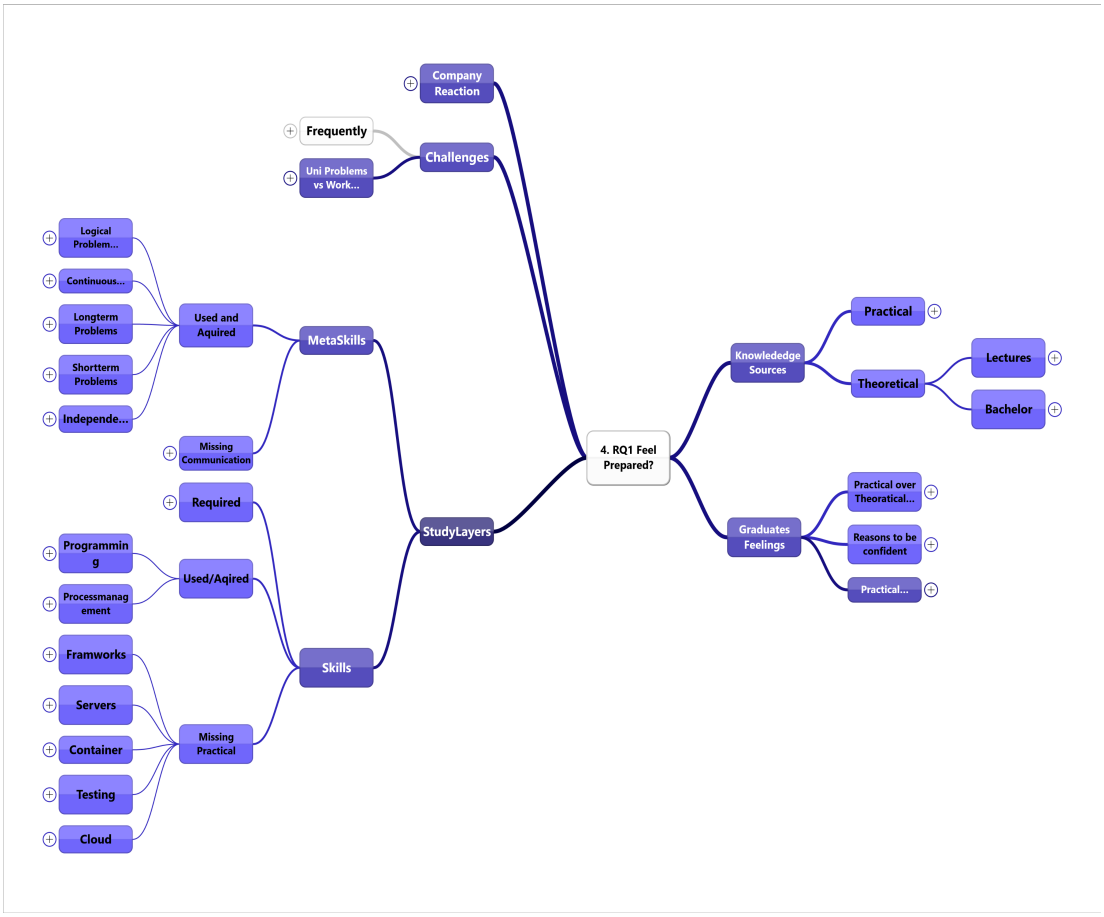


Figure 8.9: Mind map of RQ1 after the fourth Interview

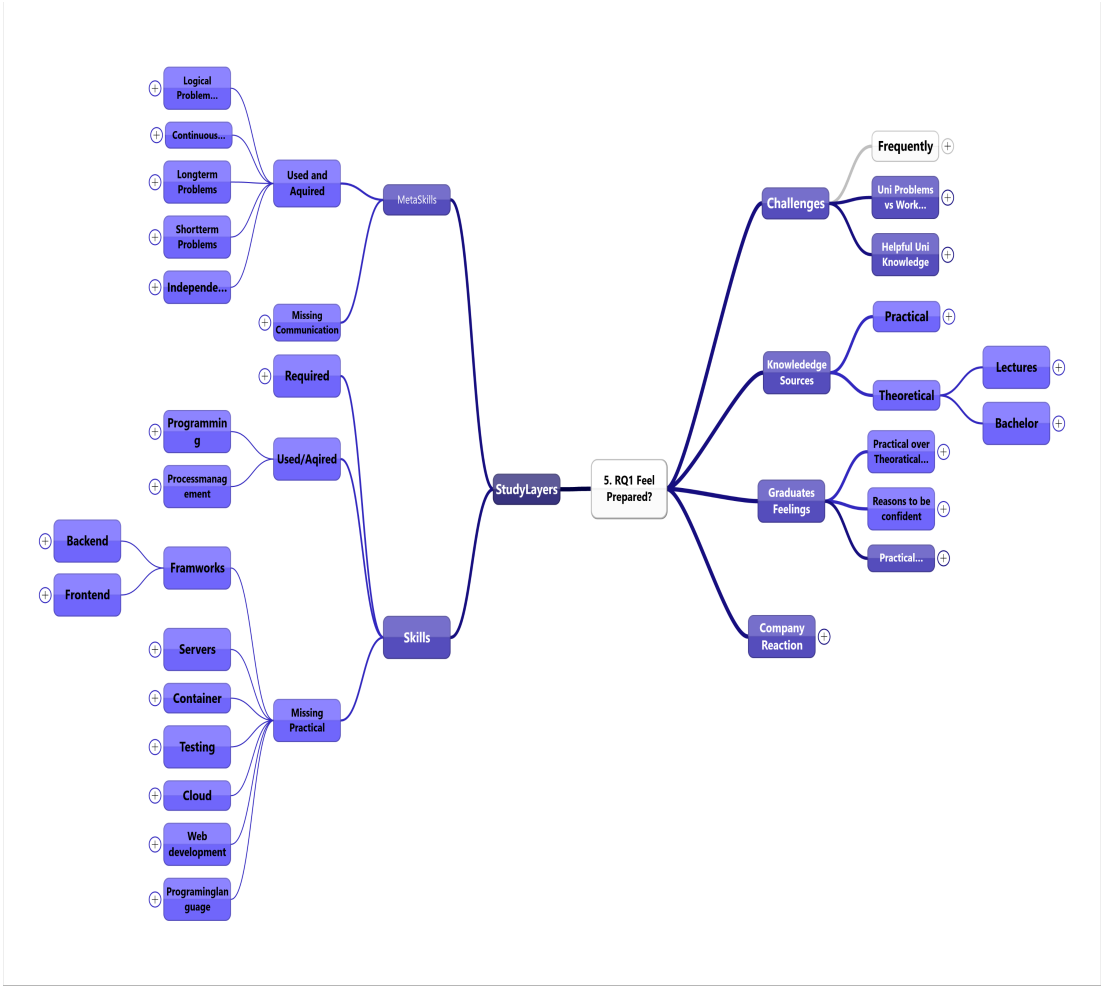


Figure 8.10: Mind map of RQ1 after the fifth Interview

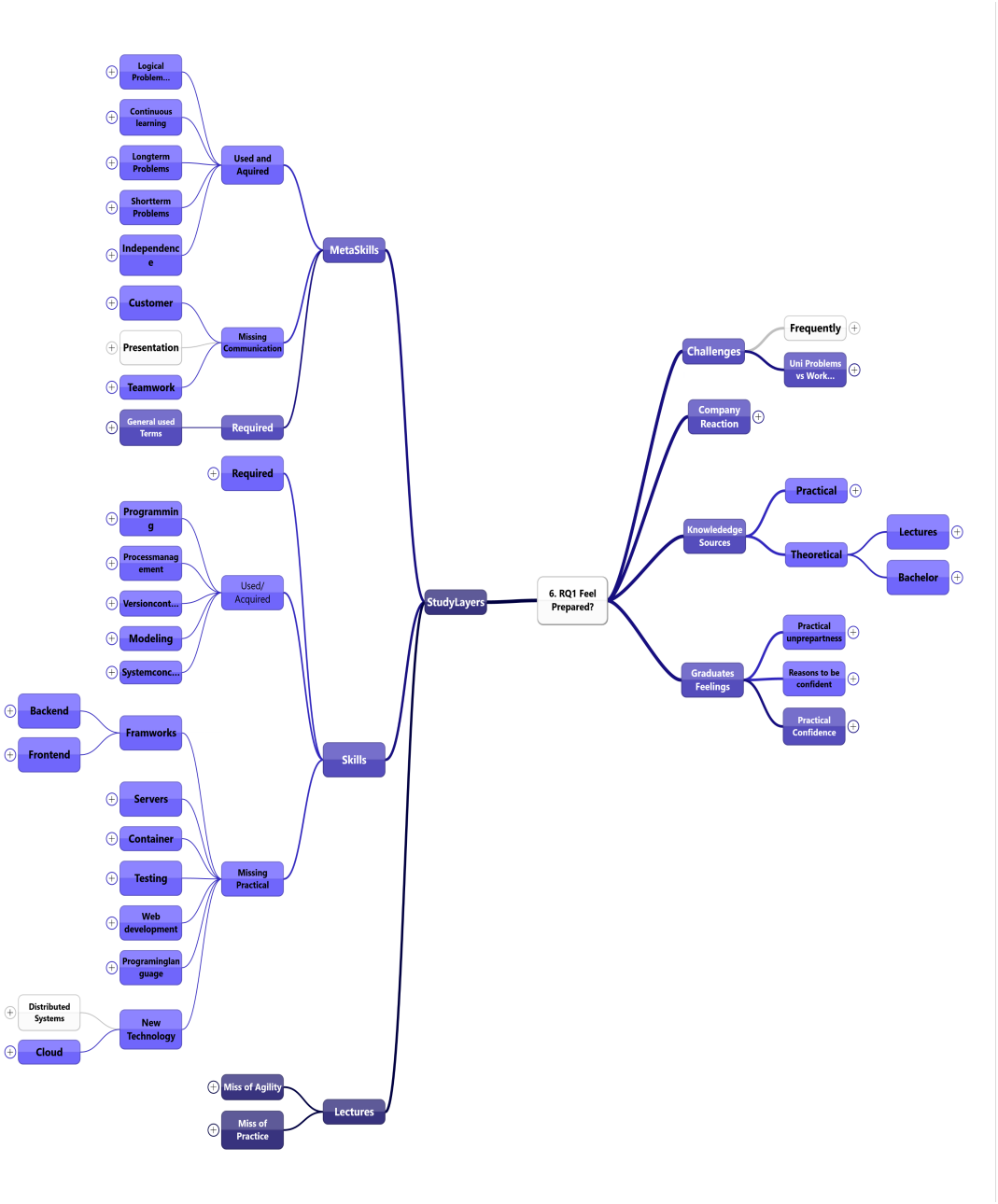


Figure 8.11: Mind map of RQ1 after the sixth Interview

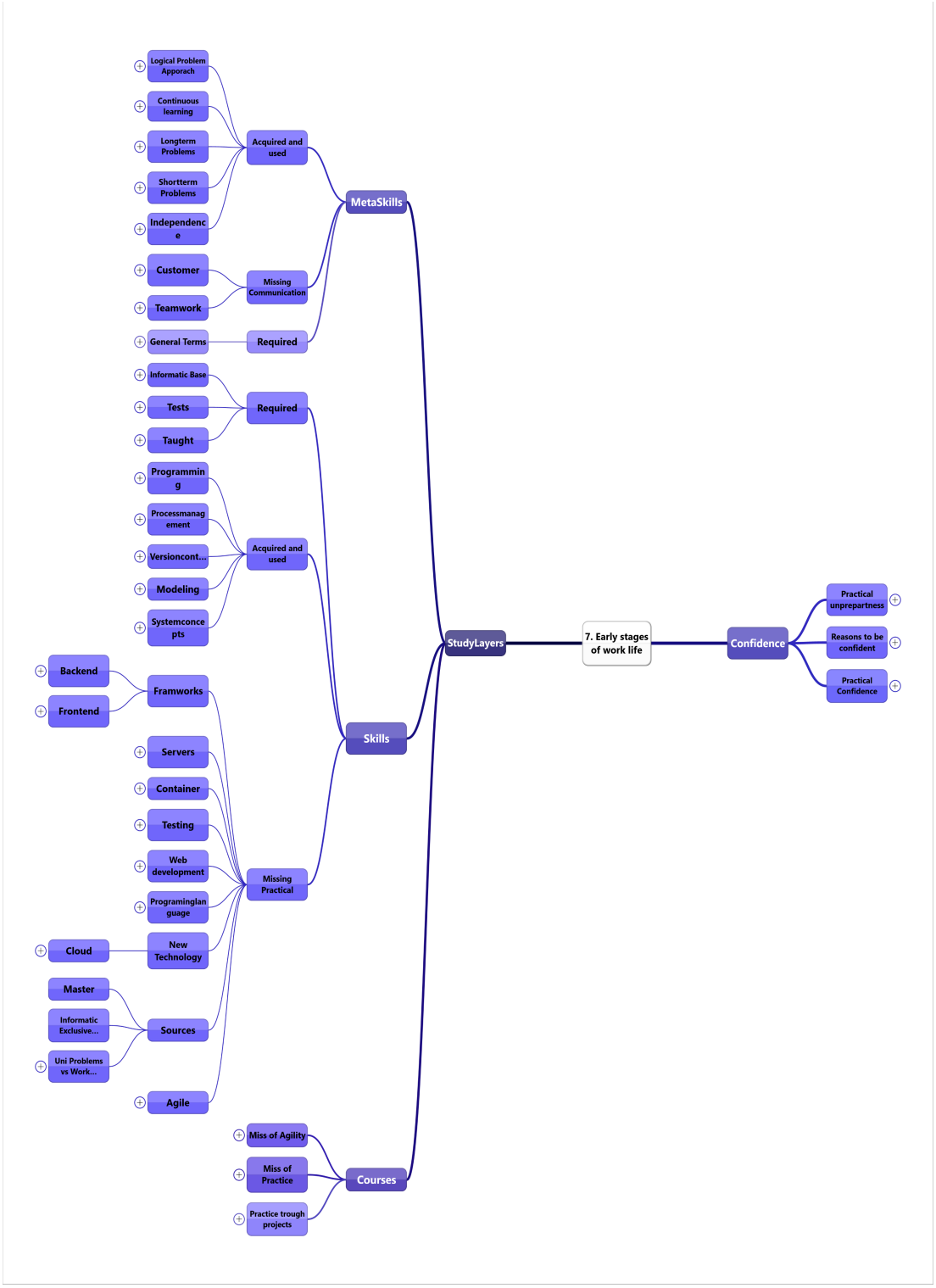


Figure 8.12: Mind map of RQ1 after the seventh Interview

9 Acknowledgement

I want to thank my supervisor here to let me choose this topic and give advice throughout the thesis process. Also big thanks to the participants who replied to me and were the base of this whole work. Everybody was really friendly and gave good answers and feedback. My Family for supporting me always for my whole life making it possible to continue my study. And my girlfriend who was there when I needed her and gave me feedback for ideas.

Bibliography

[ABCH] M. L. Alderman, M. A. Brown, M. C. Cottman, M. W. Harper. *The Australian Technology Network of Universities Referee panel* (cit. on p. 7).

[CJNZ20] O. Cico, L. Jaccheri, A. Nguyen-Duc, H. Zhang. "Exploring the intersection between software industry and Software Engineering education - A systematic mapping of Software Engineering Trends". In: *Journal of Systems and Software* (July 13, 2020), p. 110736. ISSN: 0164-1212. DOI: 10.1016/j.jss.2020.110736. URL: https://doi.org/10.1016/j.jss.2020.110736 (visited on 09/29/2020) (cit. on pp. 9, 54).

[IF19] C. Iacob, S. Faily. "Exploring the gap between the student expectations and the reality of teamwork in undergraduate software engineering group projects". In: *Journal of Systems and Software* 157 (Nov. 1, 2019), p. 110393. ISSN: 0164-1212. DOI: 10.1016/j.jss.2019.110393. URL: https://doi.org/10.1016/j.jss.2019.110393 (visited on 09/29/2020) (cit. on p. 10).

[Joc] H. L. Jochen Ludewig. *Software Engineering*. dpunkt.verlag. URL: https://dpunkt.de/produkt/software-engineering/ (visited on 05/19/2021) (cit. on pp. 13, 19).

[KC07] B. Kitchenham, S. Charters. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. 2007 (cit. on p. 9).

[LSD+16] Y. Lindsjørn, D. I. K. Sjøberg, T. Dingsøyr, G. R. Bergersen, T. Dybå. "Teamwork quality and project success in software development: A survey of agile development teams". In: *Journal of Systems and Software* 122 (Dec. 1, 2016), pp. 274–286. ISSN: 0164-1212. DOI: 10.1016/j.jss.2016.09.028. URL: https://www.sciencedirect.com/science/article/pii/S016412121630187X (visited on 06/17/2021) (cit. on p. 58).

[Max92] J. Maxwell. "Understanding and Validity in Qualitative Research". In: *Harvard Educational Review* 62 (Jan. 1, 1992), pp. 279–300. ISSN: 0017-8055 (cit. on pp. 55, 56).

[MM16] K. Matsouka, D. M. Mihail. "Graduates' employability: What do graduates and employers think?" In: *Industry and Higher Education* 30.5 (Oct. 1, 2016). Publisher: SAGE Publications Ltd, pp. 321–326. ISSN: 0950-4222. DOI: 10.1177/0950422216663719. URL: https://doi.org/10.1177/0950422216663719 (visited on 06/06/2021) (cit. on p. 7).

[MMCF05] R. Martin, B. Maytham, J. Case, D. Fraser. "Engineering graduates' perceptions of how well they were prepared for work in industry". In: *European Journal of Engineering Education* 30.2 (May 1, 2005). Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/03043790500087571, pp. 167–180. ISSN: 0304-3797. DOI: 10.1080/03043790500087571. URL: https://doi.org/10.1080/03043790500087571 (visited on 05/14/2021) (cit. on pp. 10, 55).

[Nil10] S. Nilsson. "Enhancing individual employability: the perspective of engineering graduates". In: *Education + Training* 52.6 (Jan. 1, 2010). Ed. by C. Senior, R. Cubbidge. Publisher: Emerald Group Publishing Limited, pp. 540–551. ISSN: 0040-0912. DOI: 10.1108/00400911011068487. URL: https://doi.org/10.1108/00400911011068487 (visited on 06/05/2021) (cit. on pp. 7, 10).

[PKSS17] R. Pham, S. Kiesling, L. Singer, K. Schneider. "Onboarding inexperienced developers: struggles and perceptions regarding automated testing". In: *Software Quality Journal* 25.4 (Dec. 1, 2017), pp. 1239–1268. ISSN: 1573-1367. DOI: 10.1007/s11219-016-9333-7. URL: https://doi.org/10.1007/s11219-016-9333-7 (visited on 05/11/2021) (cit. on p. 9).

[RW13] A. Radermacher, G. Walia. "Gaps between industry expectations and the abilities of graduates". In: *Proceeding of the 44th ACM technical symposium on Computer science education*. SIGCSE '13. New York, NY, USA: Association for Computing Machinery, Mar. 6, 2013, pp. 525–530. ISBN: 978-1-4503-1868-6. DOI: 10.1145/2445196.2445351. URL: https://doi.org/10.1145/2445196.2445351 (visited on 05/12/2021) (cit. on pp. 9, 54).

[SF20] K.-J. Stol, B. Fitzgerald. "Guidelines for Conducting Software Engineering Research". In: Aug. 28, 2020, pp. 27–62. ISBN: 978-3-030-32488-9. DOI: 10.1007/978-3-030-32489-6_2. URL: https://doi.org/10.1007/978-3-030-32489-6_2 (cit. on pp. 7, 14, 55).

[SGN15] M. Shah, L. Grebennikov, C. S. Nair. "A decade of study on employer feedback on the quality of university graduates". In: *Quality Assurance in Education* 23.3 (Jan. 1, 2015). Publisher: Emerald Group Publishing Limited, pp. 262–278. ISSN: 0968-4883. DOI: 10.1108/QAE-04-2014-0018. URL: https://doi.org/10.1108/QAE-04-2014-0018 (visited on 06/05/2021) (cit. on p. 7).

[SJ10] E. E. Stiwne, T. Jungert. "Engineering students' experiences of transition from study to work". In: *Journal of Education and Work* 23.5 (Nov. 1, 2010). Publisher: Routledge _eprint: https://doi.org/10.1080/13639080.2010.515967, pp. 417–437. ISSN: 1363-9080. DOI: 10.1080/13639080.2010.515967. URL: https://doi.org/10.1080/13639080.2010.515967 (visited on 06/05/2021) (cit. on p. 7).

[SOTR16] M. Sarkar, T. Overton, C. Thompson, G. Rayner. "Graduate Employability: Views of Recent Science Graduates and Employers". In: *International Journal of Innovation in Science and Mathematics Education* 24.3 (Aug. 21, 2016). Number: 3. ISSN: 2200-4270. URL: https://openjournals.library.sydney.edu.au/index.php/CAL/article/view/11043 (visited on 06/06/2021) (cit. on p. 7).

[SRF16] K.-J. Stol, P. Ralph, B. Fitzgerald. "Grounded Theory in Software Engineering Research: A Critical Review and Guidelines". In: May 18, 2016. DOI: 10.1145/2884781.2884833 (cit. on pp. 8, 11, 16, 17).

[Tym13] A. Tymon. "The student perspective on employability". In: *Studies in Higher Education* 38.6 (Aug. 1, 2013). Publisher: Routledge _eprint: https://doi.org/10.1080/03075079.2011.604408, pp. 841–856. ISSN: 0307-5079. DOI: 10.1080/03075079.2011.604408. URL: https://doi.org/10.1080/03075079.2011.604408 (visited on 06/06/2021) (cit. on p. 7).

[YAN13] S. (Yu, Amy (Chun-Chia) Chang, Natalie Tatiana Churyk. *Are Students Ready for their Future Accounting Careers? Insights from Observed Perception Gaps among Employers, Interns, and Alumni | GPAE*. 2013. URL: https://gpae.wcu.edu/index.php/2017/01/12/are-students-ready-for-their-future-accounting-careers-insights-from-observed-perception-gaps-among-employers-interns-and-alumni/ (visited on 06/06/2021) (cit. on p. 7).

All links were last followed on May 10, 2021.