Software Engineers in Transition: Self-Role Attribution and Awareness for Sustainability

Dominic Lammert
Furtwangen University and
LUT University
lado@hs-furtwangen.de

Stefanie Betz
Furtwangen University
besi@hs-furtwangen.de

Jari Porras
LUT University
jari.porras@lut.fi

Abstract

The Software Engineering process can be seen as a socio-technical activity that involves fulfilling one's role as part of a team. Accordingly, software products and services are the result of a specific collaboration between employees (and other stakeholders). In recent years, sustainability, which Requirements Engineers often paraphrase as the ability of a system to endure, is becoming part of the process and thus the responsibility of Software Engineers (SE) as well.
This study shines the spotlight on the role of the SE: their self-attribution and their awareness for sustainability. We interviewed 13 SEs to figure out how they perceive their own role and to which extent they implement the topic of sustainability in their daily work. By visualizing these two sides, it is possible to debate changes and their possible paths to benefit the Software Engineering process including sustainability design.
A discrepancy between the current role and the ideal role of SEs becomes visible. It is characterized in particular by dwelling on their "classic" or time-honored tasks as an executive force, such as coding. At the same time, they point out the still missing necessity of an interdisciplinary, from communication coined working method. According to our interviewees SEs are inefficiently involved in the design process. They do not sufficiently assume their responsibility for the software and its sustainability impacts.

1. Introduction

Meade et al. perceive the transition to agile methods at the beginning of the 21st century as one of the main reasons why the "traditional role" of the Software Engineer is dissolving. They describe the software creation process as a "complex socio-technical activity", which is no longer just about coding, but about fulfilling one's role as part of the agile group [1].

Sociology seeks to uncover how individuals behave in groups and how those groups shape their behavior.

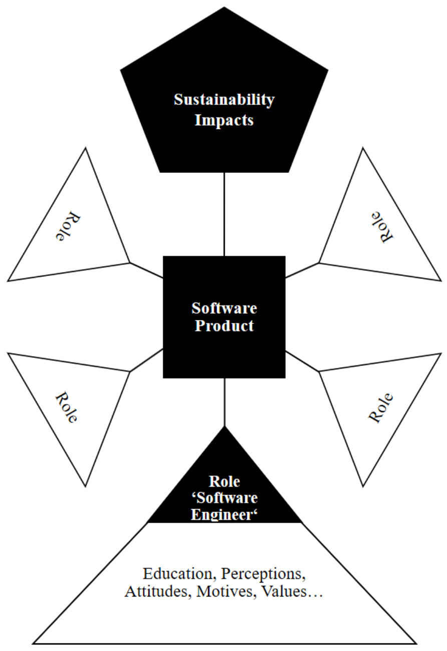

Figure 1. The software product as an outcome of the collaboration of roles and its sustainability impacts [2]

This interaction includes the formation of groups and their dynamics, as well as their maintenance and transformation [3]. If the social structures of a company change, such as the role distribution, this can have an impact on its products and services. In the last decades, an obvious paradigm shift has taken place, especially with regard to the role of the SEs. With the digital transformation, society as a whole is affected by a far-reaching change characterized by an easing of everyday and professional tasks, but also an increase in complexity due to the constant further development of existing adding new features and the ongoing emergence of new technologies, platforms and channels.
Today's SEs are no longer just an executive. SEs fulfill tasks that make them necessary in all phases of the product creation process: from design to completion [4, 5]. With the increasing involvement of SEs

URI: https://hdl.handle.net/10125/80279
978-0-9981331-5-7

in the product development process, the question of taking responsibility with regard to sustainability is also becoming louder. Software products and services can have an impact that affects social, individual, environmental, economic and technical issues. Figure 1 summarizes the process. An ensemble of roles, including that of the SE, designs and builds the software. This software in turn brings sustainability impacts .
However, it remains open at this point whether the new role definitions have sufficiently arrived in the daily work of SEs. Studies and observations show a lack of knowledge, experience and methodological support for entering the paradigm shift, especially when it comes to integrating sustainability [6]. The question arises as to whether and to what extent SEs have actually shed their traditional self-perception in order to meet the responsibilities imposed on them today. This leads to the following two research questions: **How do SEs describe their role in daily business? (RQ1)** and **What importance do SEs attach to the topic of Sustainability? (RQ2)** To lay the foundations to answer the RQs we present the background regarding roles and sustainability in the following section.

2. Background and related work

In this chapter, we lay a basic understanding of the set of topics that align the three disciplines of Sociology, Software Engineering and Sustainability. Thus, we use of a threefold structure: The concept of roles from a sociological perspective, the changing role of SE and Sustainability Design as a component in Software Engineering.

2.1. The concept of roles from a sociological perspective

At the beginning, it should be said that sociology does not offer a uniform "understanding of roles", but rather a colorful bouquet of theories. In this paper, we have chosen to rely on the most cited basic sociological works taught at universities today [3, 7, 8]. The way our social system is constructed affects each one of us in many ways. Sociological studies range from the analysis of social processes, structures, and systems to practical applications in social policy [3]. One sociological theory is that of the division of individuals into so-called "roles". The Encyclopedia Britannica describes roles as follows: "Role, in sociology, the behaviour expected of an individual who occupies a given social position or status. A role is a comprehensive pattern of behaviour that is socially recognized, providing a means of identifying and placing an individual in a society. It also serves as a strategy for coping with recurrent situations and dealing with the roles of others (e.g., parent-child roles)." [9]
When building a theoretical understanding of roles, the sociologist Erving Goffman needs to be mentioned. In his publication "The Representation of the Self in Everyday Life" [10], Goffman describes how society creates a whole range of roles and makes them available to its members. Behavior, word choice, and even clothing are symbols that are important to this role creation. They are meant to help us make the social system functional. The individual members of society classify these roles classified as "normal". Here, Goffman prefers the term "virtual identity". This is contrasted with the self-identity we have in private, where we are not under social control. Goffman calls it the "actual identity". When there is a discrepancy between our virtual identity and our self-identity, there is a risk of negative labeling that, if repeated, degenerates into stigma. Under certain conditions, roles can thus also paralyze a social system.
In contrast, we would like to add a fundamentally different sociological perspective – that of the sociologist Pierre Bourdieu. In his most widely read work "Distinction: A Social Critique of the Judgement of Taste" [11] Bourdieu recognized that people who belong to a certain class, class fraction or subclass are connected by a common taste: They prefer the same things and they reject the same things. These tastes turn out to be indicators of whether or not they "fit" into their class. Socially internalized dispositions that influence a person's perceptions, feelings, and actions are subsumed under the term "habitus". Habitus is the result of an interplay between the individual self, group culture, and social institutions. Acting out these dispositions strengthens the habitus of the individual and the group. Deviation, on the other hand, consequently causes exclusion.
The Goffman and Bourdieu approaches differ in that one suggests "people act this way" (Goffman) and the other asserts "people are this way" (Bourdieu). What both approaches have in common is that roles become socially entrenched over a period of time and, accordingly, involve a difficulty of change – whether or not a change in role is desirable. For this reason, it may be worthwhile to become aware of SE's understanding of roles. On this basis, strategies could be developed that entail an improving adjustment for collaboration in the corporate environment. In research on role understandings, the respective individual context must always be taken into account. Instead of committing ourselves to a specific role theory and applying it to SEs (inductive approach), we address the field in order to derive theories (deductive approach).

2.2. The changing role of Software Engineers

Coinciding with the changing role of SE in the 21st century, one can observe an increase in research on the connection of social sciences within the discipline of Software Engineering in numerous studies. By combining the keywords "Software Engineering" with "roles" and feed them into "Web of Science" (the world's largest publisher-neutral citation index and research intelligence platform) the increasing number of papers on this topic becomes apparent (see figure 2).

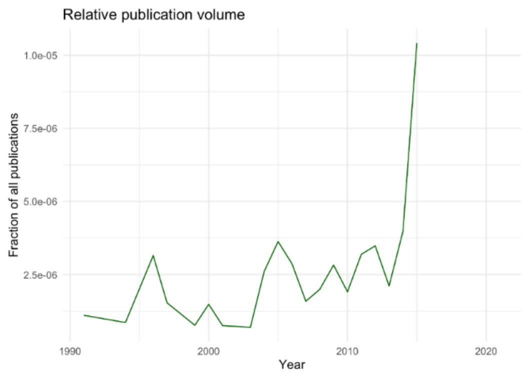

Figure 2. Output of the relative publication volume to the connection of "Software Engineering" with "roles" using "Web of Science"

It is therefore not surprising that a whole range of topics are touched upon. Michael John et al. assert that human and social factors have a significant impact on the success of software development efforts and the resulting system [12]. Tom De Marco and Tim Lister claim, "software development is highly dependent on people"[13]. Wohlin et al. explain that software development is about "balancing human, social, and organizational capital" [14]. In the following, a time-based structuring of the understanding of the role of SEs is applied, as this clarifies the paradigm shift.

Until the early 2000s: The Waterfall Model Era

The oldest publication we found which directly addresses the present topic with the title "The Role of the Software Engineer in the System Design Process" was written by Reece in 1985 in the "IEEE Military Communication Conference" [15]. The author criticizes that in design issues are decided without software expertise and that SEs should instead be involved from the beginning. The reason she gives for this is that it can reduce software implementation problems. She describes engineers as "problem solvers" and "practical people", but she also attributes management tasks to them. Three characteristics distinguish a capable SE in her opinion: technical talent, the ability to understand management concepts, and the ability to communicate.
During this time, work structures were already necessary to cope with the complexity and scope of engineering tasks, such as the waterfall model. The waterfall model consists of a sequential process in which entry into each new phase requires that the previous phase has been completed. Meade et al. state that before the introduction of today's known methodologies, projects often suffered from a lack of communication between the SEs and the users [1]. Foster recognizes another difficulty. Back then, systems were managed by the system and process analysts. As a rule, the SEs were assigned to the finance department. Most systems were correspondingly accounting-oriented, which resulted in an imbalance, which consisted in the fact that the SEs were more inconspicuous in the overall process. Their role was largely limited to execution. The work of the systems and procedures analyst on the other hand was complex, also by the fact that they had to collect the information from different departments [16]. Communication with other departments was thus less common for SEs.What also characterizes the Waterfall Model Era is that the tasks of SEs held a much narrower spectrum. They were not responsible for testing, for example, but had their own software testers. One can deduct from this that taking responsibility for their own code only became important in later years: "Software Engineers today have become more adaptable and have more responsibilities in the context of the broader project [1]."

2001 to 2011: The Agile Methods Era

The paradigm shift was apparently initiated in the year 2001, when SEs published the "Manifesto for Agile Software Development" [17]. It stands for a profound review and reorientation of the practice of Software Engineering. Meade et al. cite that as software development has evolved, the role of a SE has become "broader and more heterogeneous." Technological advancements had resulted in a change in the role of SEs and changed the needs of companies. This development is not standing still, as new technologies – the authors cite artificial intelligence and machine learning as examples – are constantly being added [1].
Open source software likewise contributed to the paradigm shift by creating the path to communities of collaborators and contributors on a variety of projects [1]. Today's employees can no longer limit themselves to their so-called department and what they were "classically" trained to do. The change in job titles, the renaming of departments and the repartitioning of

organizations attest to this shift [18]. Foster attributes to today's SEs that they serve in an advisory capacity to the entire organization, and they are a change agent who advocates (and implements) system improvements from a wide variety of viewpoints. In doing so, they must be aware of all planned organizational changes that relate to the software system they develop [16].
At this point, it can be observed that the role of SEs is becoming more versatile. New areas of responsibility, in which they have to familiarize themselves depending on the project, are added. (Interdisciplinary) communication is gaining in importance. A SE has a say in the design of the software.

Since 2011: The Start up Era (or Agile Methods 2.0)

Based on that, the start-up scene and its mentality brought an additional level of agility to the stage. The next level agile methods are characterized by iterations, as can be seen for example in the "Lean Startup Circle" by Eric Ries. It is about constantly building, measuring and learning where Business Economists, Marketers, SEs and other stakeholders need to put their heads together strategically as team members to move the company forward [19]. In addition, a differentiation of a psychological component regarding SE can be seen in current publications. The number of types of SEs is immeasurable. Feldt et al. detect correlations between SE personality views and attitudes [20]. Soomro et al. find an influence of SEs' personality traits on team climate and performance [21]. Karimi et al. seek to understand the influence of a bundle of personality factors on programming styles and performance [22].
Capretz and Ahmed offer a mapping of soft skills and psychological traits to the main phases of the software lifecycle to reflect the complexity and importance of the topic. For this mapping, the researchers used the Myers-Briggs Type Indicator (MBTI), a well-known instrument for measuring and understanding individual personality types. They conclude that software developers are "introverted (I), feeling (S), thinking (T) types." [23] Gorla and Lam explain that the variation of personalities in a project can have a balancing effect in the workplace [24].
SEs today contribute to a large extent to the design of the software, which was not a matter of course in the past. Depending on how much importance is attached to the topic of sustainability, software has different positive and negative impacts for which SEs share responsibility.

2.3. 2014 until our-days: The Software Sustainability Era

We would like to start this section with a famous quote from Grady Booch: "every line of code has a moral and ethical implication." [25] This statement requires an explanation.
According to a study of Wolfram et al., the increasing concern with climate change as well as a growing awareness of social inequality have led to the topic of sustainability being accorded increasing importance overall, from which Software Engineering is not unaffected. In 2017, the researchers set up a systematic mapping study with the aim of identifying where and how the issue of sustainability is being addressed in Software Engineering. To this end, they evaluated 1035 studies on the topic of sustainability and green IT [26].
In 2014, Becker et al. established the "Sustainability Principles for Software Engineering" in the so-called "Karlskrona Manifesto for Sustainability Design" [27], which can be considered the starting point for the current era. As can be seen from the website www.sustainabilitydesign.org, the authors and signatories of the manifesto (software practitioners and researchers) write that their intention is to align concern for the planet and society with Software Engineering. According to them, the narrative about sustainability and the role it plays in the profession of SEs, among others, needs a redefinition. The work of SEs is accompanied by a responsibility regarding sustainability impacts of the software systems they design that they have to face. The signatories establish a broad understanding of the term "sustainability" by pointing out its five correlated dimensions:

- **Social:** includes relationships between individuals and groups.
- **Individual:** includes the ability of individuals to flourish, exercise their rights, and develop freely.
- **Environment:** includes the use and management of natural resources.
- **Economic:** includes the financial aspects and business value.
- **Technical:** includes the ability of the technical system to adapt to change.

Furthermore, a distinction is made between three different effects: **immediate effects** (start with the production, use, and disposal), **enabling effects** (arise over time), and **structural effects** (changes on the macro level that alter our society). This is also the definition of sustainability as it applies to software for

purposes of this research.
In the same year, Betz et al. introduced the concept of "sustainability debt". This metaphor, borrowed from economics, is intended to help discover, document, and communicate sustainability issues in Requirements Engineering: "Sustainability matters for all software systems, even if the application domain of the system is not related to sustainability, because any new software creates dependencies and obligations as it becomes part of our technical infrastructure, and its on-going use may entail new burdens on social and ecological systems." [28] SEs usually focus on technical issues. However, software systems also affect non-technical systems. Only at a second glance we perceive their societal, environmental, and economic interactions. SEs should not abdicate this responsibility [27].
In 2016, Becker et al. agreed to this circumstance by claiming that the social role of software, which is often considered critical, necessitates a paradigm shift in the mindset of SEs. The authors explain that designing for sustainability poses a major challenge. Complex software-intensive systems influence sustainability in the five correlated dimensions. In terms of sustainability, SEs would have to adopt a mindset different from that of the puzzle solver. Rather, they now face "wicked problems", or problems that are entrenched in a complex system. Responsibility can only be sufficiently taken into account if there is an awareness of sustainability [27].
Oyedeji et al. present concepts that can be used to evaluate green and sustainable software systems. This includes measurement of the five software sustainability dimensions [29]. On the other hand, in 2016, Chitchyan et al. addressed the relevance of sustainability in Software Engineering discipline while emphasizing that there is little knowledge about how it is perceived by SEs and, as a result, how sustainability design can become part of the design process. The 13 respondents in this study only associated sustainability with the availability of natural resources and the reduction of waste, only with the environmental dimension. There is a lack of knowledge and therefore awareness of the other four dimensions [30]. The measurement methods thus still need to be further developed and established.
2019, Duboc et al. stated that software occupies every component of social life (from commerce, communication, education, to energy, entertainment, finance, governance, and defense, etc.), making socio-technical systems a key factor in sustainability [31]. As recently as 2020, however, Duboc et al. emphasized that Requirements Engineers lacked the knowledge, experience, and methodological support for this task. For several years, various tools have been developed to help Requirements and SEs consider sustainability in the software development process [6].
In summary, the number of publications in this thematic field is likewise growing. We can divide these as follows. First, scientists publish the basics in sustainable Software Engineering and call for awareness and responsibility. Second, they teach methods that help to take account of the "sustainability debt". Thirdly, they carry out studies that examine software companies according to their approach to the issue of sustainability. The topic of sustainability is still quite new in Software Engineering, so that a formalization of sustainability as part of the Software Engineering "have yet to make their way into official standards and models" [26]. Going back to Fig. 1, the software is the result of collaboration between different roles. The software in turn leads to sustainability impacts. Scientists worldwide agree that SEs must take their responsibility into account more strongly than has been the case to date. We concur with the scientific findings cited in this section and proclaim that SEs integrate the five dimensions of sustainability as well as its three effect levels into their work.

3. Research gaps

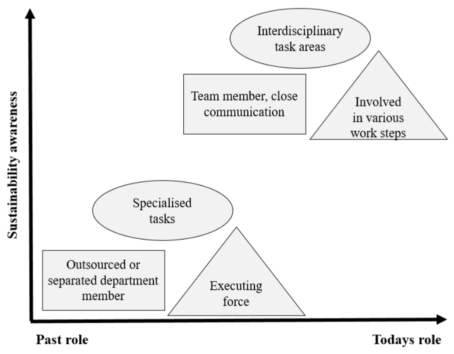
Sustainability awareness
Interdisciplinary task areas
Team member, close communication
Involved in various work steps
Specialised tasks
Outsourced or separated department member
Executing force
Past role
Todays role

Figure 3. Self-role attribution and sustainability awareness of SEs in the past and today

Putting the last chapter together, we can say that over the past decades, the role of SEs has been subject to numerous upheavals. Until the transition from the "Waterfall Era" to the "Agile Methods Era", they move closer and closer to colleagues from other areas with whom they formed teams. Specific tasks that made them the executive force of companies became interdisciplinary task areas that integrated them in different work steps. This inevitably increased the responsibility for the product. SE can no longer be measured only by the quality of their code; they also bears responsibility for the design of the overall project. This also applies to the responsibility for

sustainable Software Engineering. Figure 3 summarises this development.
One of the questions that has not yet been adequately answered is how the transition actually looks like in practical implementation. It is about the given circumstances in companies towards the combination of the self-role attribution of SEs on the one hand and on the other hand their awareness for sustainability. The purpose of this explorative study is to get a closer look at the actual state of software companies in order to check to what extent the current status corresponds to this transition. This study serves to provide initial answers to this research gap in order to build a bridge for the development of follow-up studies.

4. Interview study design

This study is part of a broader investigation on the connection between Software Engineering and sustainability design. Here, we conducted an exploratory qualitative research by conducting 13 interviews with SEs. Within the qualitative research guidelines, we followed scientific rules in the field of interviews summarized by Elmer [32]. We did not ask the SEs directly for a description of their self-role attribution, nor did we ask them directly for a definition of sustainability. If we had asked for these two, there would be a risk that the answers would correspond to social desirability. The first part of the interview was about the SEs' profession in general and their role in the company. Here it was interesting to see if the interviewees bring in the term sustainability themselves and if any of the described activities can be related to sustainability. The second part of the interview was about the impact of software. Here, too, the interviewees were to be as free as possible in describing their view of things, the ”status quo”, when it comes to impacts. We have chosen a semi-structured form in order to achieve a flexible survey, although it can be time-consuming to conduct and evaluate and it also means that the number of respondents is rather small. On the other hand, it is a recommendable form of research to ensure understanding and to obtain extensive statements.
In the **planning phase**, we opted for a deductive sample, since there is of course already knowledge about the people who can provide information about the question. The respondents should be employees whose job title falls under the term “Software Engineers”. To remain constant, we left it at this company size (including those who consider themselves as startups) for all interviews. We do not intend to aim for representativeness for the “typical” SE.
We conducted half of the **data collection** through face-to-face interviews and the other half through voice over IP platforms. Twelve software practitioners worked in Germany and one in France. All interviewees have been working in their professional field for more than one year: between 2 and 22 years. The average was 7.7 years. We made sure to cover different industries to create versatility. The 13 interviewees belong to the following industries: Finance (2x), IT Security (2x), Web and App Development (2x), Big Data, eCommerce, Energy, Environment, Language Learning, Marketing, and Social Media. Five of them had a close connection with universities in their daily work. Four of the SEs received a one-year scientific start-up grant, which provides guidance from a university professor as well as the relevant university start-up office. Two startups are based on a business concept developed during proseminars at the university. Here, there are contacts to the former lecturers of these seminars as well as to student founder initiatives, which support the respective startup with know-how and networking. One of the startups was a university project in which several universities are involved in the founding. The respondents were all male and their age ranged from 29 to 55 years. They all had a diploma, master's or bachelor's degree. All interviews we carried out in English. They were recorded and transcribed (anonymised). The implementation took place in the months of May to September 2020. The interviews took about an hour each.
In the **data analysis**, to structure the responses, we developed an open coding strategy. Two researchers read through each of the interviews, coded them, and additionally peer-reviewed each other's work to establish a codebook on which all agree. For qualitative content analysis, we selected the approach of a deductive category application. We used text analysis software as a tool here. The results of this interview study fall into two categories: self-role attribution and sustainability awareness.

5. Study findings

In this chapter, we present our most important results and findings. We divide the study findings in two parts: Self-role attribution and Awareness for Sustainability.

5.1. Self-role attribution

With the first part, we want to get an insight into how SEs would describe their role within the company itself. We asked this question directly: “How would you describe your role as a software developer within your company to others?” Next, we asked them to describe a regular working day and what they thought would make an ideal working day. Explicitly, we asked about the skills and competences a SE needs. Finally, we asked about the advantages and disadvantages of integrating the SE into the overall product development process. The answers to these questions should contribute to the overall picture of the role. We started coding job description, which led us to a bundle of tasks mixed up with required skills and competencies. This includes a list of codes, which constitute a regular and an ideal working day in the eyes of the SEs. Here, the relationship and a distinction from other stakeholders became clear. The interviewees also answered by letting us know about framework conditions or the working environment. Finally, we created two code lists that we divided into the opportunities and the risks that lie in the integration of the SEs into the entire product creation process: from generating an idea until the market entry.

Tasks, skills and competencies. The SEs describe their areas of responsibility as diverse. We divided the tasks, and thus the required skills and competencies, into five areas based on the respondents' answers: Technology, Communication, Project Management, Finance and Others.

The focus is clearly on the first area and its complexity: Technology. In total, the interviewees count on 25 different technical tasks when it comes to their regular work. It turned out that it is not enough to go into everything in an interview lasting around an hour. One interviewee stated: "So my role in my company is to develop projects and software from A to Z." An idealization of programming skills is particularly evident. Terms such as "efficient code", "good coding", "satisfying outcome of code", "clear code", "clean code" and "working code" are used. The respondents describe a comprehensive knowledge and understanding of programming languages, software architecture and the system as equally important.

The communication area appears second most frequently. They address it in both the current and the ideal role conception. Discussions, mutual understanding and efficient communication with meetings and digital communication (chats and e-mails) are clearly considered relevant.

Project management tasks are in third place. Activities such as organising, planning and coordinating tasks are standard in this profession. This aspect also applies to some SEs in the fourth area: Finance. One interviewee speaks of the relevance of a "business-oriented mind", which SEs should ideally have. Other tasks, skills and competencies differ from company to company. It can be about generating ideas, learning, presenting, holding workshops or even campaigning and political work.

Relationship with other stakeholders. One of the most obvious things in the entire evaluation is that SEs distinguish themselves from their other colleagues by identifying themselves as the "doers", as those who "get things done": "In my experience, the developer is at least the one who is doing the things, who has to get the things done. And many times he has a lot of people around him who are talking, and planning, and organizing money and other political things. But finally the core product is my daily work, is my job. And yeah, so I have to be the one who gets things done."

This narrative is often encountered, also concerning the ideal SE: "[...] someone who is really focusing on getting stuff done." Accordingly, clear tasks are ideally expected: "I think really important is to have a clear task that we can focus on." This circumstance does not coincide with the responsibility that SEs have to participate in the product design and thus also in the accompanying task orientation. Some of them prefer to be given tasks that they then only have to carry out rather than participating in defining them.

In contrast, the interviewees often mentioned the term "teamwork" as well as their relationship to other "professional groups": product / project owner, customer, designer, facilitator, marketer, project manager and scientist. SEs have to sit at the table here because (unlike the other groups of colleagues) they knows what is feasible and what is not. Here is one example about the connection between the designer and the SE: "[. . .] the designer knows what the best practices are for example, or best workflows with some interfaces, but the programmer knows what is doable and what is, according to that time and that amount of money, this project tests, what is doable." Therefore, they also mentioned social skills. One interviewee emphasizes that the software is the result of teamwork. "And then also there's this cliché of software developers of being the weird nerds down in the basement, and that is totally wrong. It does have that part as well, but it's not like this is 90 percent. [...] It's always teamwork. And so, the team aspect and the social aspect is very important for communicating problems and getting things done well."

Working environment. The interviewees did not mention much about the general situation of the working environment. Some said that they do not have a regular working day and the term "home office" came up, which can be attributed to the consequences of the regulations due to the current Corona situation.

More interesting was what they made known about their ideal ideas of a work environment. They made statements that can be put under the heading of "undisturbed work". One interviewee explained: "Not to be disturbed by other colleagues or customers, who can work normally, straight way." Another one found similar words: “Room for silence” and “room for concentration” are necessary to “dive in and be un-disturbed and work for like three hours on some development topic.” The need for privacy can be transferred to programming, because only in this activity is the presence of other colleagues not needed. Seen in this light, the question arises as to whether SEs do not need two rooms: a common one for exchange and a sole one for execution.

Integration throughout the design process. The advantages of involving SEs in the entire product development process clearly outweighed the disadvantages. They were primarily recorded in bringing in a technical perspective, to “find the compromise between this perfect design and what’s really doable.” Additionally, answering financial questions also played a role here: “So, I think software developers generally should have an entrepreneurial mind and understand what is the value of that what they are building?” A third reason lies in the enrichment through a different way of thinking and working: “I think the developer should be part of the whole process because they bring in another point of view, a technical point of view.” The disadvantages referred to the fact that the greatest strength, the technical focus, can degenerate. SEs tend to slow down processes because they become rigid on technical aspects: “I would say the biggest threat for a self-motivated developer is working too long on unnecessary things. A classical problem would be early optimization or working on features that nobody requests. That is, I think the biggest problem.”
Another threat is that they are not involving themselves into the discussion: “And software development is sitting on the site is listening, is not saying something.” One respondent mentioned the lack of communication skills by stating that someone is necessary “who can speak as well the language of developers”.

5.2. Awareness for sustainability

The second part of the questions we started by asking about the use of tools and frameworks. Our intention was to check whether the topic of sustainability is addressed here on its own. We then asked if they also use tools or frameworks that address questions about ethics, consequences or sustainability. Afterwards, we addressed questions about the importance of such issues within the company. Finally, the questions focused on the integration of these issues into the training of SEs.
Not a single SE interviewed uses ethical frameworks or tools as a guidance. Twelve of them have never heard of the ACM Code of Ethics and Professional Conduct, only one of them knew the term.
When it comes to moral issues within their work, eleven of them raise the issue of data security. The other dimensions of sustainability and their effect levels (see section 2.3) do not come up. Two SEs claim not to encounter any moral issues. From this, a truncated understanding of sustainability could be derived. Four SEs attach importance to ethics in their work, compared to nine who do not share this view. In the case of the importance of impacts, nine attribute an importance to the topic. Two see an importance here, but classify it as rather low. Two others do not assign any importance to impacts.
Accordingly, with nine, the majority of SEs thinks about impacts when creating a product or service. In contrast, four interviewees answered this question in the negative. Nine of the interviewees are also of the opinion that the topics dealt with here should be integrated more strongly into Software Engineering.
The question of whether SEs should be trained differently in the future was mostly not addressed in terms of ethical questions or questions about impacts and sustainability. Six of the interviewees answered this question in the negative. Of these, four claim that it is more about personal issues. Another interviewee claimed that SEs need to acquire knowledge on these topics themselves. In addition, one stated that SEs do not need to be strong in every area. Three interviewees stated that they had absolved a training in these topics during their training. In contrast, there were ten for whom this was not the case. Only one of the SEs needs help in dealing with these issues. Five of them showed openness or interest. Seven answered in the negative.

6. Discussion

RQ1: We can conclude that the SE has evolved into a team member who communicates closely with other team members, who covers interdisciplinary task areas and who is involved in various work steps. However, we cannot say that the SE does yet seem to have fully arrived in this new role. The focus lies on technology and seems to be so strong and the interdisciplinary way of working so low that non-technical tasks in communication become more difficult and even being perceived as disturbing. SEs have not yet completely dissolved their role as a purely executive force. They continue to see themselves in the role of doers who need clear tasks.
RQ2: At the same time, SEs shy away from their responsibilities, which can be seen in particular in the sustainability consequences. The application of methods regarding sustainability is not part of their toolbox; the consequences are mainly seen in data

security. However, this does not mean that they are not aware of sustainability issues. With nine interviewees, the majority stated that these topics should be integrated more strongly into the Software Engineering process than before. They were not usually trained in this topic, but half of them brings up a motivation to catch up (six out of thirteen). This can be seen as motivation to tackle the issue.Standards are required as well as an expansion of the scope of activities in order to meet the sustainability debt.

Overall, there are still uncertainties in the "right" way to deal with this issue. We know from sociology that roles solidify over time and that it is not possible to dissolve and transform them overnight. As far as sustainability design as a component in Software Engineering is concerned, we are also dealing with a relatively new research topic. Developments with regard to both research questions can certainly be identified, but the desired theory and the real practice still diverge significantly.

7. Limitations

We have conducted a qualitative study thus there are a number of aspects threating the validity of our findings. We have considered these systematically, discussing the four threats to validity: construct validity, internal validity, external validity, and reliability.

Construct validity: A threat to construct validity may be that interviewees may not understand the questions, and the interviewer may misinterpret data. To minimize this threat, we ensured that the interviewees had sufficient experience in Software Engineering; further on, to provide a context for some of the questions, we asked the interviewees to read a small part of the ACM Code of ethics before the second stage of the interview started. Furthermore, we piloted the interview to make sure that the questions were clearly stated and answerable. Moreover, the interviews were taped allowing the researchers to listen to the interviews again to limit misinterpretation. Lastly, coding was then conducted pairwise. Another threat to construct validity is reactive bias to the presence of a researcher. To reduce that threat, interviewees have been assured their anonymity and we use open questions in the interviews as a way to reduce interviewer bias. Also, an interview guideline had been agreed upon the three authors and followed after the first pilot interview.

Internal validity: To minimize the impact of confounding factors influencing the analysis we applied qualitative analysis techniques. Additionally, we do not claim that we collected any other data but that for practitioners perceptions and attitudes related to their work practices and to sustainability, and how these may shift when an ideal working situation is considered. However, threat of confounding factors cannot be ruled out completely.

External validity: The cases presented here are not statistically representative and are not intended to as this is a qualitative study, and statistical generalization is not our goal. Our explorative, qualitative study was designed to help us identify the perceptions of the interviewees with regard to their roles, their responsibilities and possible to enable sustainability design. By selecting practitioners from different application domains, and company sizes, we focused on the collection of a rich set of data.

Reliability: To minimize threats to reliability, coding was done pairwise. Any mapping disagreements were discussed until consensus was reached.

8. Conclusion and future work

If we look at software companies through a sociological lens, we see that every employee has a role to play in order for the software to be completed. The role of SEs has changed since the early 2000s with the rise of Agile Methods. Ideally, SEs no longer limit themselves to specialized tasks, but to interdisciplinary task areas. Today, they can no longer be described as an executive force alone, as they are and must be involved in numerous work steps. They work less in outsourced or separated departments and more in close teams with a high degree of communication. For some years now, we have been able to observe that the topic of sustainability is becoming a component that can only be implemented with the help of SEs. For this development to gain momentum, it is essential that SEs receive a sense of responsibility for the fact that their work has an impact on five dimensions: social, individual, environmental, economic and technical. In addition to publications on Sustainability Design to raise awareness, researchers are publishing methods to help meet the sustainability debt. With both topics, self-role attribution and awareness of sustainability, it is evident due to some discrepancies that the path from former to today's desired structures has not been completed. SEs are insufficiently involved in the design process because their focus on technical issues is so strong that there are communication difficulties with team colleagues from other areas. Their sense of responsibility is thus also on the technical side, such as whether the code works adequately. Sustainability concepts intended for the software design process do not adequately take into account the fact that SEs lack sustainability awareness and a general sense of responsibility for their Software.

This is limited to the issue of data security.

For future studies, we consider two areas to be relevant. First, we plan to complement our qualitative approach quantitatively to ensure that our findings reveal a broader problem rather than an isolated one. A qualitative study such as this entails aspects that threaten the validity of the results. Second, the existing sustainability design methods that are already in use in the Software Engineering process should be reviewed. In this way, the feedback from the participants can be analyzed so that readjustments can be made. These adjustments should take into account the SEs self-role attribution as well as their sustainability awareness.

References

[1] E. Jones Meade, E. O'Keeffe, N. Lyons, D. Lynch, M. Yilmaz, U. Gulec, R. O'Connor, and P. Clarke, *The Changing Role of the Software Engineer*. Springer, 2019.

[2] D. Lammert, "The connection between the sustainability impacts of software products and the role of software engineers," *Evaluation and Assessment in Software Engineering (EASE) Doctoral symposium*, 2021.

[3] C. Thorpe, C. Yuill, and M. Hobbs, *The Sociology Book*. DK Publishing, 2015.

[4] I. Sommerville, *Software Engineering*. Pearson, 2019.

[5] R. E. Bourque, Pierre; Fairley, *Software Engineering Book of Knowledge (SWEBOK)*. IEEE Computer Society Press, 2014.

[6] R. Chitchyan, C. Venters, and et al., "Requirements engineering for sustainability: An awareness framework for designing software systems for a better tomorrow," *Requirements Engineering*, vol. 25, p. 469–492, 2020.

[7] P. S. Adler, P. Gay, G. Morgan, and M. Reed, *The Oxford Handbook of Sociology, Social Theory and Organization Studies*. Oxford University Press, 2014.

[8] H. Griffiths, N. Keirns, and E. e. a. Strayer, *Introduction to Sociology*. OpenStax, 2015.

[9] T. E. of Encyclopedia Britannica, ed., *Role*. Encyclopædia Britannica, Inc., 2020.

[10] E. Goffman, *The Presentation of Self in Everyday Life*. Doubleday, 1956.

[11] P. Bourdieu, *Distinction: A social critique of the judgement of taste*. Routledge, 1984.

[12] M. John, F. Maurer, and B. Tessem, "Human and social factors of software engineering: workshop summary," *ACM SIGSOFT Softw. Eng. Notes*, vol. 30, no. 4, pp. 1–6, 2005.

[13] T. DeMarco and T. Lister, *Peopleware: Productive Projects and Teams (3rd Edition)*. Addison-Wesley Professional, 3rd ed., 2013.

[14] C. Wohlin, D. Šmite, and N. B. Moe, "A general theory of software engineering," *J. Syst. Softw.*, vol. 109, p. 229–242, Nov. 2015.

[15] M. J. Reece, "The role of the software engineer in the system design process," in *MILCOM 1985 - IEEE Military Communications Conference*, vol. 2, pp. 346–349, 1985.

[16] E. C. Foster, *Software Engineering: A Methodical Approach*. Appress, 3rd ed., 2014.

[17] K. Beck, M. Beedle, A. v. Bennekum, and et al., *Manifesto for Agile Software Development*. 2001.

[18] I. Alexander and S. Robertson, "Understanding project sociology by modeling stakeholders," *IEEE Softw.*, vol. 21, p. 23–27, Jan. 2004.

[19] E. Ries, *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Sydney: Currency, 2011.

[20] R. Feldt, L. Angelis, R. Torkar, and M. Samuelsson, "Links between the personalities, views and attitudes of software engineers," *Inf. Softw. Technol.*, vol. 52, p. 611–624, June 2010.

[21] A. B. Soomro, N. Salleh, E. Mendes, J. Grundy, G. Burch, and A. Nordin, "The effect of software engineers' personality traits on team climate and performance," *Inf. Softw. Technol.*, vol. 73, p. 52–65, May 2016.

[22] Z. Karimi, A. Baraani-Dastjerdi, N. Ghasem-Aghaee, and S. Wagner, "Links between the personalities, styles and performance in computer programming," *J. Syst. Softw.*, vol. 111, p. 228–241, Jan. 2016.

[23] L. F. Capretz and F. Ahmed, "Making sense of software development and personality types," *IT Professional*, vol. 12, no. 1, pp. 6–13, 2010.

[24] N. Gorla and Y. W. Lam, "Who should work with whom? building effective software project teams," *Commun. ACM*, vol. 47, p. 79–82, June 2004.

[25] G. Booch, *Software engineering in practice keynote: The future of software engineering*. ICSE'15: Proc. of the 37th Intl. Conf. on Software Engineering, 2015.

[26] N. Wolfram, P. Lago, and F. Osborne, "Sustainability in software engineering," in *2017 Sustainable Internet and ICT for Sustainability (SustainIT)*, pp. 1–7, 2017.

[27] C. Becker, S. Betz, R. Chitchyan, L. Duboc, S. M. Easterbrook, B. Penzenstadler, N. Seyff, and C. C. Venters, "Requirements: The key to sustainability," *IEEE Software*, vol. 33, no. 1, pp. 56–65, 2016.

[28] S. Betz, C. Becker, R. Chitchyan, L. Duboc, S. Easterbrook, B. Penzenstadler, N. Seyff, and C. Venters, "Sustainability debt: A metaphor to support sustainability design decisions," *CEUR Workshop Proceedings*, vol. 1416, pp. 55–63, 2015.

[29] S. Oyedeji, A. Seffah, and B. Penzenstadler, "Classifying the measures of software sustainability according to the current perceptions," in *MeGSuS@ESEM*, 2018.

[30] R. Chitchyan, C. Becker, S. Betz, L. Duboc, B. Penzenstadler, N. Seyff, and C. C. Venters, "Sustainability design in requirements engineering: State of practice," in *2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C)*, pp. 533–542, 2016.

[31] L. Duboc, S. Betz, B. Penzenstadler, S. A. Koçak, R. Chitchyan, O. Leifler, J. Porras, N. Seyff, and C. C. Venters, "Do we really know what we are building? raising awareness of potential sustainability effects of software systems in requirements engineering," in *27th IEEE International Requirements Engineering Conference, RE 2019*, pp. 6–16, IEEE, 2019.

[32] S. S. Elmer, "Mündliche befragung (oral questioning)," in *Empirical scientific work. A study guide for the educational sciences* (J. Aeppli, L. Gasser, E. Gutzwiller, and A. Tettenborn, eds.), vol. 4, pp. 177–191, UTB, 2016.