Analysing a Systematic Literature Review Combined with an Undergraduate Survey on Misconceptions about Software Engineering

Carolin Gold-Veerkamp
University of Applied Sciences Aschaffenburg
Aschaffenburg, Germany
Email: carolin.gold-veerkamp@th-ab.de

***Abstract*—From a constructivist perspective, learning is an active, cognitive process in which individuals construct their own knowledge by connecting new concepts with previous knowledge, skills, and experience that serve as points of departure. The purpose of this study is to identify and analyse misconceptions in Software Engineering to use these insights for higher education. Therefore, a systematic literature review as a secondary data accumulation as well as a primary data acquisition covering undergraduates are conducted, analysed, and compared. Concerning the SLR, out of 2,158 publications found, only 20 evidence-based misconceptions/beliefs/myths from 3 papers could be found. The undergraduate survey resulted in 69 misconceptions covering 13 interviews. Additionally, both approaches have been compared.**

***Keywords*—Software Engineering; Higher Education; Misconceptions; Systematic Literature Review; Undergraduate Survey.**

I. Introduction

From a constructivist point of view, learning is to be understood as an active, individual, situated, social, and cognitive psychological process. Each individual has to build up their own knowledge by combining new concepts based on previous knowledge, existing competencies, previous experience, as well as conceptions and putting it into a network-like relationship. This means, learners form conceptions and models to explain phenomena, processes, and artifacts before they are confronted with them in institutional learning. These possibly alternative – from scientific or expert perspective – conceptions have a twofold significant impact on the learning process. On one hand, they can serve as the basis for learning, on the other, they can also contradict the educational content and thus hinder the learning process.

To be able to do so in Software Engineering (SE) education, we have to take a step back and clarify which misconceptions undergraduates bring to university. Thus, this contribution is based upon two pillars, a Systematic Literature Review (SLR) as a secondary data analysis [1] as well as a primary data acquisition [2] that both provide information about SE misconceptions. These shall be presented, compared and contrasted.

As a consequence, the goal is to achieve sustainable learning (in higher education), a purely technical structuring of the learning content is therefore insufficient. Furthermore, didactics should do justice to the learners' "points of departure" [3, p. 6]. The Model of Educational Reconstruction, which is epistemologically based on the constructivist position, calls for precisely this consideration [3] [4]. The model comprises the triad of *content clarification*, *learners' conceptions*, and *didactic design*; it considers the scientific concepts and the student conceptions as equivalents.

This contribution is structured on the basis of these two pillars: The first part covers the SLR (Sections II-IV) starting with terminological aspects, as a plurality of terms evolved, and related work in other disciplines. The SLR process is explained in Section II, complemented by the results (Section III) as well as a short discussion (Section IV). The second part explaines the survey process (Sections V-IV) covering the approach in Section V, the results (Section VI) and again a short discussion in Section VII. The third part (Section VIII) combines both pillars by analysing and comparing the MCs found; the aim is not to conduct a one-to-one matching, but to look on context-related proximity.

A. Terminological Aspects

Due to many different ways of looking at the research object '*(mis-)conceptions*' as well as the critical examination of the terminology, an abundance of terms has developed. The different understandings have led to a plurality of terms with multiple connotations. The abundance of technical terms has risen so much in the course of research (especially in natural sciences didactics) that it is now almost impossible to survey. The fact that the terms cannot be clearly distinguished from each other often leads to a more or less synonymous use and thus to an undifferentiated mix. As a result of the dissatisfaction with this situation, researchers have again constructed and defined additional terms, which expands the existing term dilemma.

In addition to [5] [6], also others include collections of terms. This list (merely referenced by several publications) gives an impression of the broad spectrum:

- *Preconceptions* [5]–[8]
- *(Students') conceptions* [5] [6]
- *Alternative conceptions* [5]
- *Naïve conceptions* [6]
- *Naïve theories* [5] [6]
- *Naïve beliefs* [5]
- *Beliefs* [9]
- *Alternative beliefs* [5]
- *Alternative frameworks* [5] [6] [8]
- *Intuitive theories/science* [8]
- *Prior knowledge* [8]
- *Misconceptions*, the "standard term" as [5, p. 119] state – despite the negative connotation [6] [8].

In spite of the heterogeneity of terms, opposed opinions and discussions on the different types of expression, it can be stated consensually that individuals each develop different conceptions of certain concepts, which should and must be used as a starting point in teaching. These conceptions can, but do not have to be in line with modern scientific theories [6] and therefore may act as learning obstacles [10], often referred to as *misconceptions* (MC).

B. Related Work on Misconceptions in Didactics

The research and publications about misconceptions in natural sciences in the context of school are immense, as a bibliography by Duit [11] proves. When looking at the catalogue, encompassing over 8,300 publications and summarizing them per decade (Figure 1), it is obvious that since the mid-1970s international researchers have been investigating the field.

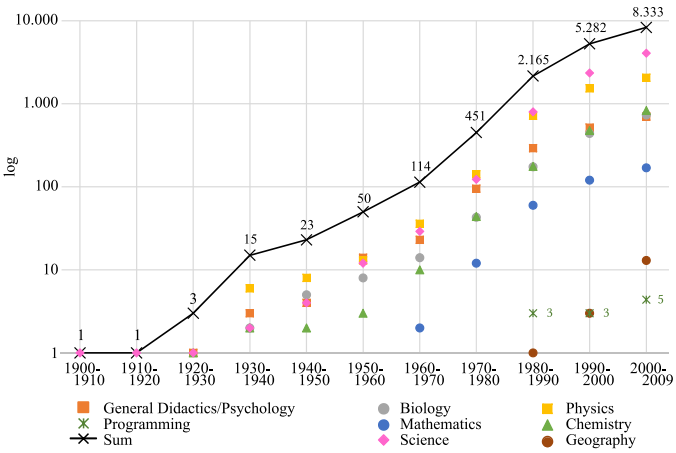

Fig. 1. Diagram of Accumulated Number of Publications per Decade, Sorted by Discipline Listed in [11]; esp. Focused on 'Programming'

Out of these, merely five publications [12]–[16] can be assigned to 'programming' as nearest to SE, but also to science and/or maths; i.e., they are equivocal.

Moreover, in the last few years, several papers on misconception research in computer sciences appeared concerning:

- ... programming [17]–[20] and object-oriented programming in particular [21] [22].
- ... artifacts, e.g., computers, smartphones, and so on [23, and others].
- ... the Internet [24].

All publications listed have in common that they do not particularly deal with SE and are contextualised within school education. This results in research needs for SE in university.

II. Method: SLR on Misconceptions in SE

In order to be able to present the state of research on (undergraduate) conceptions in SE, there is a need for a SLR, which summarizes all available information about this phenomenon thoroughly and impartially [25, p. 7]. Conducting a SLR is a quantitative methodology of secondary data collection for the synthesis of research results from primary studies. The guidelines – used here – that Kitchenham and Charters have drawn up for SE are derived from several approaches in medicine and the social sciences [25, p. vi].

The following explanations, which describe the three-phase process of the SLR – carried out as a computer-based, automated search – are divided into the initial planning in Section II-A, the actual practical implementation (Section II-B) and the subsequent presentation and use of the results (Section III), see also [25] [26].

A. Phase 1: Initial Planning

The planning of the SLR contains some parameters that require previous definition in order to minimize bias. The SLR is determined as follows:

1) Research Question(s): To what extent does research on misconceptions in SE already exist? Which misconceptions in SE are known/documented?

2) Search Strategy:

a) Language Selection: At this point, the language radius, which is one of the inclusion criteria, should be anticipated. The reason for this is the following definition of the Search Query (SQ). Since research on conceptions is international, publications in German and English are considered.

b) Queries and Synonyms: Regardless of the various connotations (Section I-A), the search should encompass the previous research on MCs in SE as broadly as possible. Therefore, the search query is based on the numerous synonymously used English terms shown in Table I. (Indicating wildcards, i.e., placeholders, by an asterisk (*).)

TABLE I
Development of the English Search Query Using Synonyms

Synonyms	Substrings	
	Noun	Adjective
preconceptions	preconception*	–
students' conceptions	conception*	–
alternative conceptions		alternative
naïve conceptions	–	naïve
naïve theories		
naïve beliefs	belief*	–
beliefs		alternative
alternative beliefs	–	
alternative frameworks	–	intuitive
intuitive theories	–	
intuitive science	–	prior
prior knowledge	misconception*	–
misconceptions		

In contrast to *preconception*, *conception*, *belief* and *misconception*, the terms *theory*, *framework* and *science* (plus plurals) are only included in combination with the respective adjectives (Table I), since they are often used as technical terms in SE and unspecific for answering the research question. Same applies to the terms *student*, *knowledge* and *science*, because of the usage of pedagogical databases. These are combined with the disciplinary focus on SE, resulting in Search Query 1; including wildcards (*) and search for exact phrases (quotation marks). (The equivalent German SQ is not attached here.)

c) Database: Electronic literature databases are selected based on Kitchenham et al. [27] in combination with [28]. Kitchenham et al. have already dealt intensively with SLRs in the area of SE and set up a list of important English-language journals and conferences, which they themselves use for their literature research (see Table II).

(“software engineer*” OR “software development*” OR “software process*”)
AND
(“preconception*” OR “conception*” OR “belief*” OR “misconception*” OR
“naïve theor*” OR “alternative framework*” OR
“intuitive theor*” OR “intuitive science” OR “prior knowledge”)

Query 1. English Search Query

TABLE II
SELECTION OF ELECTRONIC DATABASES FOR SLR BASED ON [27] [28]

Source	IEEE	ACM	SD	SC	SL	ERIC	WoS	GS	arXiv	dblp
Information and Software Technology			X	X				X		
Journal of Systems and Software			X	X				X		
IEEE Transactions on SE	X							X		
IEEE Software	X							X		
Communications of the ACM		X								
ACM Computer Surveys		X								
ACM Transactions on SE Methodologies		X								
Software Practice and Experience								X		
Empirical SE Journal					X					
IEEE Proc. Software (now: IET Software)	X							X		
Proc. Int. Conference on SE	X	X						X		
Proc. Int. Symp. of Software Metrics	X	X						X		
Proc. Int. Symp. on Empirical SE	X	X						X		

These are used as a basis to identify databases that include these compilations, namely: IEEE-Xplore [29], ACM-Digital Library [30], SpringerLink (SL) [31], Scopus (SC) [32], and Science Direct (SD) [33]. This selection is supplemented by further search engines from the educational context (ERIC [34], Web of Science (WoS) [35]) and the metadata database GoogleScholar (GS) [36]. In addition to the proposed ones, arXiv [37], an open access repository for electronic preprints from numerous areas – including computer science –, and the dblp [38], which is co-founded by the German federal government, are used.

3) Selection Strategy: The selection is controlled on the basis of the following predefined Inclusion (IC) and Exclusion Criteria (EC).

IC.1 The publication is written in English or German language.

IC.2 It is explicitly about the discipline SE.

IC.3 MCs in SE are explicitly mentioned.

EC.1 The contribution is an abstract, workshop, poster, or similar, as these do not provide in-depth information.

4) Quality Assessment: The gathered publications have to be qualified against predefined Quality Criteria (QC):

QC.1 *Traceability*: How do the authors know this misconception? It is scientifically important to be able to track where the information comes from.

QC.2 *Validation*: Has it been confirmed that it is a misconception? How did the authors validate the conception to be “at odds with modern scientific theories” [6, p. 2]? If not done, there is no indication that it is really a *mis*conception.

QC.3 *Occurence in the population*: Does this misconception exist in the population? Did the authors test the misconception in a specific target group? Otherwise, the existence of the misconception is not empirically proven at all or limited to individual subjects (e.g., through interviews).

B. Phase 2: Conducting the SLR

The process of conducting the SLR is shown in Figure 2 as *Phase 2* of the overall process.

1) Stage 1: Conducting the Automated Search: For the search – if possible – use of extended/advanced search functions, wildcards (e.g., “misconception*”), and Boolean operators is made in order to be able to exploit the predefined syntax of the query (see String 1). Nevertheless, the string must be adapted to the options of the search engine. Care is taken to ensure no semantic changes take place.

The SQ is limited to document title and abstract, as recommended by [39, p. 2050] as well as others. (Deviations from this definition, due to the search options of the individual databases, are documented accordingly in the evaluation in Section III). The reason for this is that both metadata are already indicators of the relevance of a publication. Note: At this point, IC.3 is not completely applicable, since MCs are not specifically mentioned in title & abstract, but it is checked, whether the contribution is explicitly about misconceptions.

2) Stage 2: Applying the In-/Exclusion Criteria: The relevance of a publication is determined in a two-stage process (see Figure 2, *Stage 2*). First of all, the title and abstract are examined and evaluated on the basis of the predefined criteria. These provide enough information to decide whether a publication encompasses insights of interest; in doubt they were included. The papers included are then rechecked regarding the in-/exclusion criteria; this time considering the full text.

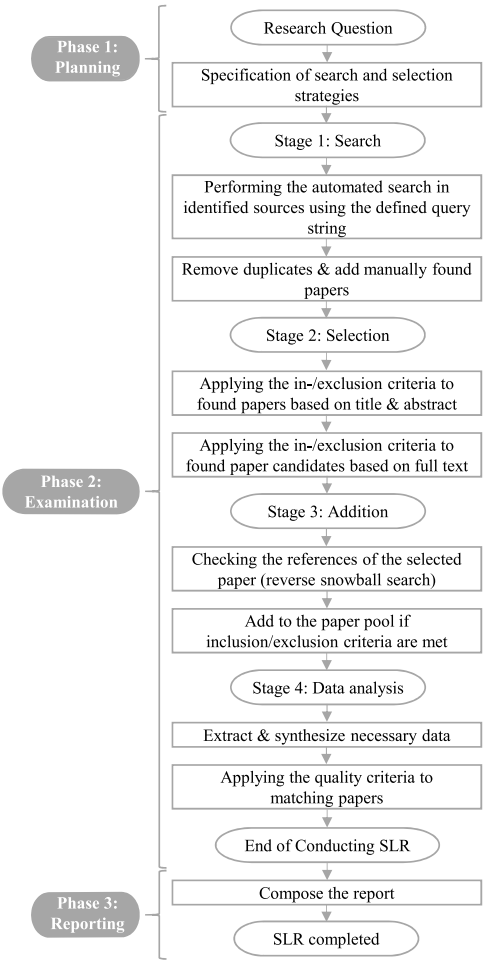

Fig. 2. Flowchart of the SLR process (based on [25] [26] [39])

3) Stage 3: Backward Snowballing: Once *Stage 2* is completed, "the references of the selected papers [are] reviewed and any missing candidate papers [are] assessed against the inclusion/exclusion criteria" [39, p. 2052] as well; this is referred to as 'backward snowballing'.

4) Stage 4: Data Analysis: To assess the quality of the methods and results in the gathered publications, quality criteria have to be predefined against which to assess the data extracted and synthesized.

III. Phase 3: Results of the SLR

The results of the coarse search based on the selection criteria (Section II-A3) applied to titles and abstracts (Section III-A) and the detailed search using full texts (Section III-B) are presented. Additionally, the results of the analysis of the MCs found in the selected publications is shown in Section III-C, which is based on the QCs (Section II-A4).

A. Results: Coarse Search

The automated search has been completed between April, 30th and May, 1st 2020. Since the search was not limited to a date range, the review process timewise included every research found, covering papers as of 1970. Table III illustrates the number of matches ($n = 2,158$) initially received through the SQs. Excluding data sets that contained entire proceedings/compilations instead of contributions as well as duplicates, results in $n = 1,481$. Finally, after applying the inclusion and exclusion criteria to title and abstract, $n = 128$ papers/articles can be identified as potentially relevant to our interest. Therefore, only these are considered in the next step, in which the full text of these publications is considered.

Duplicates could be localized both internally – within the results of the same SQ, within the same database, or overlaps between English and German SQs – and externally – between the results of different search engines. The number of duplicates can be seen in Table IV including multiple mentioning, as papers might be found in multiple databases. (Therefore, the sums are not equivalent with the numbers of duplicates in Table III.)

TABLE IV
Number of Duplicates

	IEEE	ACM	SD	SC	SL	ERIC	WoS	GS	arXiv	dblp
IEEE	15	32		222		1	2	15	5	12
ACM		54		111				9	4	7
SD			18	60				4		2
SC				53		7	4	31	10	21
SL					0					
ERIC						3		3		
WoS							0	3		4
GS								17		19
arXiv									8	2
dblp										4

TABLE III
Summary of SLR Results After Applying In-/Exclusion Criteria on Title & Abstract

	Search Engines										Sum
	IEEE	ACM	SD	SC	SL	ERIC	WoS	GS	arXiv	dblp	
Results of English SQ	250	410	93	847	0	29	4	87	257	41	2,018
Results of German SQ	16	54	7	46	0	3	0	7	2	5	140
Sum of Search Results	266	464	100	893	0	32	4	94	259	46	**2,158**
No Papers (e.g., Proc.)	2	2	0	53	0	2	0	34	6	0	99
Duplicates	15	85	18	383	0	10	4	16	18	29	578
Balance without Duplicates	249	377	82	457	0	20	0	44	235	17	1,481
IC.1a: English	249	352	81	442	0	20	0	40	231	10	1,425
IC.1b: German	0	0	0	5	0	0	0	1	0	0	6
IC.2: SE Discipline-Specific	223	253	65	381	0	18	0	34	162	7	1,143
IC.3: Misconceptions	30	60	4	43	0	8	0	6	5	2	158
EC.1: Contribution Type	0	1	0	0	0	1	0	0	0	0	2
EC: No Information	0	0	0	1	0	0	0	2	0	0	3
Paper Candidates	**29**	**40**	**4**	**37**	**0**	**6**	**0**	**5**	**5**	**2**	**128**

TABLE V
SUMMARY OF SLR RESULTS AFTER APPLYING IN-/EXCLUSION CRITERIA ON FULL TEXTS

	Search Engines										Sum
	IEEE	ACM	SD	SC	SL	ERIC	WoS	GS	arXiv	dblp	
Paper Candidates (see Table III)	29	40	4	37	0	6	0	5	5	2	128
IC.1a: English	29	40	4	37	0	6	0	5	5	2	128
IC.1b: German	0	0	0	0	0	0	0	0	0	0	0
IC.2: SE Discipline	29	40	4	37	0	6	0	4	5	2	127
IC.3: Mention Misconceptions	5	3	1	2	0	0	0	1	3	0	15
Papers Found	**5**	**3**	**1**	**2**	**0**	**0**	**0**	**1**	**3**	**0**	**15**
Backward Snowballing	27	5	0	2	0	0	0	1	4	0	39
Already Included in SLR	2	1	0	1	0	0	0	0	2	0	6
After Applying Selection Criteria	0	0	0	1	0	0	0	0	2	0	3
Result	**5**	**3**	**1**	**3**	**0**	**0**	**0**	**1**	**5**	**0**	**18**

B. Results: Full Text Search

Proceeding further, the predefined inclusion and exclusion criteria are then applied to the paper candidates based on the full text of the contributions. This results in $n = 15$ papers that match the criteria (see Table V). Papers are excluded that cover the topic 'misconception', but did not explicitly mention at least one statement the respective authors refer to as an MC concerning the topic SE (cf. IC.3). The subsequent backward snowball search – based on the adequate papers found – reveals some additional publications that have been checked against the inclusion/exclusion criteria listed as well. Summing up, a total of $n = 18$ papers are found (see Table V) that are of interest to the research question of this SLR.

Through the selection process in *Stage 2* and Backward Snowballing in *Stage 3* as a whole, we double-checked the contributions by assessing each paper. As Kitchenham et al. suggest, publications are included if we cannot make a consensual decision [39, p. 2052].

The matching papers found ($n = 18$, shown as the result in Table V) are listed below:

- IEEE: [42]–[46]
- ACM: [47]–[49]
- Science Direct: [50]
- SCOPUS: [51] (cites and covers the myths of the primary source [52] and 7 new statements) [52] [53]
- Google Scholar: [54]
- arXiv: [55] [56] (is included in [41] and thus not considered further) [40] (is the basis for [41]) [41]; and thus considered together, covering 21 MCs in total) [57]

C. Results: Misconceptions Found

Within the publications named, a total of 167 individual statements (see Table VI; without cross-references) are declared as misconceptions by the respective authors. The MCs gathered should be evaluated by assessing the quality of the publications in order to determine the capacity of the findings, using the quality criteria from Section II-A4.

The coding of the subcategories of the quality criteria was not determined in advance, but developed during the analysis based on and close to the available data; i.e., the publications themselves. The following subcategories are considered as high-quality (see grey marking in Table VI):

QC.1 *Traceability*: A primary *study* as well as the *reference* to quotable publication(s), in which the MC(s) were found is defined as satisfying scientific claims. In contrast, no indication is insufficient.

QC.2 *Validation*: The conception has to be *empirically confirmed* as "at odds with modern scientific theories" [6, p. 2] to be a *mis*conception. Whereas, a rejection, an explanation by the author(s) or reference(s) that the statement given is supposed to be a misconception is no sufficient evidence for validation. This is also due to the fact that MCs exist in all ages, from primary

TABLE VI
SUMMARY OF MISCONCEPTIONS FOUND IN THE FULL TEXTS USING THE QUALITY CRITERIA

		Papers Found																	Sum
		[38]	[39]	[40]	[41]	[42]	[43]	[44]	[45]	[46]	[47]	[48]	[49]	[50]	[51]	[53]	[54]	[55]	17
Misconceptions explicitly named		16	12	12	7	6	4	5	12	4	7	7	10	36	4	10	21	4	167
QC.1: **Traceability**	- Study						4							36		21 (in [40])			51
	- Reference(s)	15	6			6													37
	- No Indication	1	6	12	7			5	12	4	7	7	10		4			4	79
QC.2: **Validation**	- Empirically Confirmed								12							8 (in [41])			20
	- Empirically Rejected															2 (in [41])			2
	- Reference(s)	6	6		5														17
	- Only based on Explanation						4			4	7	7	10		4				36
	- No Indication	10	6	12	2	6		5						36		11		4	92
QC.3: **Occurrence**	- Practitioners	16				6	4									21 (in [40])			37
	- Undergraduates		12	12					12										36
	- No Indication				7			5		4	7	7	10	36	4	10/21 (in [41])		4	94
Intersection (of rows marked)									12							8			20

TABLE VII
LIST OF MISCONCEPTIONS MATCHING THE QUALITY CRITERIA

Topic(s)								Misconception	Reference(s)
Project	Process Models	Team Skills	Reuquirements	Implementation	Defects	Documentation			
X								A defined software process is only important when you are working with people who are less skilled.	[49, (1)]
	X							A good software developer will often choose to work alone on a project in order to get it done faster.	[49, (2)]
X	X							When you have a team of good programmers who work well together, a software process will usually get in the way.	[49, (3)]
			X					My code should take advantage of the implementation details in other code.	[49, (4)]
		X						It is expected that clients will describe their requirements accurately before a team begins programming.	[49, (5)]
				X				As a software developer, most of my time will be spent designing and implementing new algorithms and data structures.	[49, (6)]
X								Most of the time when I start a new programming task in industry, I will be working on a new project.	[49, (7)]
				X				Developers do not need to know the high-level context of the system; this allows them to concentrate on their task.	[49, (8)]
					X			A software project is successful only if it ships with very few known defects.	[49, (9)]
						X		Software engineering is about producing lots of documentation on the requirements and implementation of the project.	[49, (10)]
	X	X	X					Process, requirements, and team-management are important to business majors, not software developers.	[49, (11)]
				X				The majority of the cost of a successful software project will be the initial implementation effort.	[49, (12)]
					X			A file with a complex code change process tends to be buggy.	[40, (S2)], [41, (B1)]
					X			A file that is changed by more developers is more bug-prone.	[40, (S14)], [41, (B2)]
					X			A file with more added lines is more bug-prone.	[40, (S4)], [41, (B3)]
					X			Recently changed files tend to be buggy.	[40, (S7)], [41, (B4)]
					X			Recently bug-fixed files tend to be buggy.	[40, (S10)], [41, (B6)]
					X			A file with more fixed bugs tends to be more bug-prone.	[40, (S11)], [41, (B7)]
					X			A file with more commits is more bug-prone.	[40, (S12)], [41, (B8)]
					X			A file with more removed lines is more bug-prone.	[40, (S13)], [41, (B9)]

level to university and even experts and professors can hold them themselves [58, p. 9, 11].

QC.3 *Occurrence in the population*: Practitioners MCs are included, as it is very likely that students have them as well, if they can be encountered in professionally experienced. However, no indication of occurrence in the population can initially only be interpreted as a presumption.

The intersection of the QCs results in $n = 20$ misconceptions (Table VI). Yet, the papers [40] and [41] only deal with the topic 'defect prediction', the authors of [49] look at SE covering the software life cycle more holistically; see thematic structuring in Table VII.

Note: [49] would actually not be included in the intersection, as it is not explained where the MCs come from (QC.1). But the authors validated them (QC.2) and tested their occurrence concerning students (QC.3). Thus, the MCs listed are hypotheses, that have been empirically confirmed; thus, nevertheless, they are included in the intersection.

IV. DISCUSSION

Several aspects regarding the SLR should be remarked upon concerning the validity of the methodology as well as the results per se.

A. Methodology: Threats to Validity

First, one significant limitation is the broad number of synonyms for 'misconception'; it is almost impossible – despite all efforts – to ensure that all relevant papers are found.

Second, we used the four-eyes principle to proceed and discussed to achieve consensus, but enclosed papers causing persistent disagreement. However, this is not an ideal process, affecting reliability of assessment and evidence of results.

Third, a limitation is that own publications could turn out to be matches in the SLR, which must be handled objectively. This can result in a systematic error. It is therefore noted that authors of this paper also authored the publication [54].

B. Discussion of Results

Regarding the results of the SLR, it is noted that the cut of 2,158 publications to merely 3 [40] [41] [49] of interest identified is immense. As a result, it could be assumed that the search (engines/query) or the selection (in-/exclusion/quality criteria) are inadequate. However, this contradicts that ...

- ... SE didactics are still developing.
- ... the consideration of another database (Section I-B, [11]) also indicates that little research is available to date.
- ... other authors report the same for the adjacent field of computer sciences: "At present, hardly any empirical data concerning the issue of expectations and prior knowledge [...] in informatics [...] are available" [59, p. 143].

V. APPROACH: UNDERGRADUATE SURVEY ON MISCONCEPTIONS IN SE

This section covers the methodology to detect Misconceptions through GTM (Sec. V-A) and downstream to construct Misconception items for the catalogue (Sec. V-B).

A. Detection of Misconceptions through GTM

After presenting the process of conducting the SLR, the following sections explain the the second pillar of this contribution – the survey carried out: its design, the process as well as the results.

1) Design of the Undergraduate Survey: Due to several reasons and criteria, the decision was made to operationalise Straussian GTM and apply it in a tailored implementation [60] to detect undergraduates Misconceptions in SE as a step to build upon. The main causes are the explorative nature due to the research gap detected (see Section III; [1], interpretative reconstructing from the statements, and openness towards the data to grasp the undergraduates' understanding. The Grounded Theory (GT) Methodology can be described as an iterative process that is constantly enriched by new data. This was done after a careful interpretation why a qualitative,

explorative and data-driven research approach and GTM, in particular, seems to be a promising research approach regarding the research goal and the research already available. Furthermore, a review of the three main variants of GT (Glaserian [61]/Straussian [62]/Constructivist GT [63]) has been done to determine relevant points of similarity and difference, to define the best solution possible for this application scenario. This has also been done since Stol et al. demonstrated the "method slurring" [64] that GTM is suffering from through a critical examination of the usage of GTM in SE [60].

The central research question of the GT study therefore is [65]: What Misconceptions do undergraduates have concerning Software Engineering?

2) Data Elicitation and Analyses: The study was carried out as follows with reference to key concepts of GTM:

a) Theoretical Sensitivity: This concept [66] has an ambivalent significance to the research process. On the one hand, contextual knowledge is essential to enable the researchers to localize concepts in and abstract them from the data. On the other hand, the assumptions and results may harbour the risk of a bias. Due to the academic background of the researchers[1] and due to a previously conducted small literature review on (Mis-)conceptions research in SE (in order to detect and formulate the research gap; see [65]), there is a high theoretical sensitivity on the present study. However, maybe in contrast to other research approaches, the SLR shown in Sections II-IV was not conducted in advance (cf. Section V-A2b). In order to deal with this tension field, this information has been disclosed here. This transparency as well as a reflection/discussion on this possible problem source is a way of disarming it, using traceability and verifiability of the findings.

b) Data Collection: Undergraduates' statements about SE have been elicited using interviews as a form of primary data acquisition. To recruit the individual interviewees, project members from different Universities of Applied Sciences – professors as well as research associates – set up the contact. The 16 interviews (14 male/2 female) conducted took place between June 14, 2017 and December 12, 2018.

c) Theoretical Sampling: Since GTM follows simultaneous data collection and analysis, the sample selection is driven by *Theoretical Sampling*. In this study, different criteria have been used to implement minimum and maximum contrast, which evolved through the research progress: University or place of study, degree program, and study progress.

As seen from Table VIII, the interviews have been conducted in waves of small cohorts (N = 3...6), which – per wave – represent intra-similar data (minimum contrasting). Due to the study purpose, it is not always possible and useful to progressively collect data, but at certain times several interviews are conducted en bloc, because of a specific point in the semester that should be captured; e.g., in the beginning, the focus was on the target group entering the module(s) in SE. Interviewees were recruited through announcements made in courses by lecturers (colleagues at the different Universities of Applied Science). Participation was voluntary.

[1]The researchers received Master degrees in computer sciences as well as engineering. Furthermore, they have worked in discipline-based educational research for several years as research associates.

TABLE VIII
UNDERGRADUATE INTERVIEW PARTNERS IN THE SAMPLE

Wave	N	University	Degree Program[a]			Study Progress
			CS	MT	EE&IT	
1	4	C	X			2nd/3rd Semester[b]
2	3	A		X		2nd/3rd Semester[b]
3	3	C	X			Freshman
4	5	B			X	Freshman
	1			X		
Sum	16		7	4	5	

[a] CS = Computer Sciences, MT = Mechatronics, EE&IT = Electrical Engineering and Information Technology

[b] The undergraduates have been asked in advance to module(s) dealing with SE, which take place in the 3rd/4th semester, depending on the degree program, but have already gained some programming experience and other basics.

Despite their divergent characteristics, they stated similar statements that could be reconstructed as Misconceptions, concerning a strong focus on programming, testing, and greatly simplifying requirements engineering. This gave rise to the question: Is the reason for this possibly to be found in the previous programming training they took part in? Therefore, the decision was made to interview freshman students as well and contrast their statements to the already established code system (encompassing Misconceptions).

By contrasting the individual's data with interviews of similar and dissimilar interviewees, the theoretical epistemological value could be maximized. This constant comparison led to a theoretical saturation, which was reached when, despite additional data, no new information emerged from the data.

d) Guided Interviews: Data acquisition took place using guided interviews that encompassed questions like:

- *Have you already gained experience in the field of "Software Engineering" outside of your studies?*
- *How is software actually created? / How do you imagine software being developed? / What does this process look like?*
- *What happens in the development process until the software is finished?*
- *When is a Software Engineer's job done?*
- *How are the tasks divided into a project in terms of time?*
- *Which people are involved?*
- *What skills should a "good" typical Software Engineer have? Why?*

The undergraduates have been interviewed individually. The interviews lasted between 15 and 44 minutes (average = 23.75; SD = 7.05) depending on the undergraduates' openness towards the interview situation, readiness to leave their personal comfort zones, willingness to speak, and the number of further inquiries by the interviewer to meet the scope of the interview guideline and get a perception of the interviewees understanding. At the beginning of each interview, undergraduates were explicitly assured that this is not a test, but rather a feedback to get to know their conceptions and ideas in order to be able to adapt the courses. The interviews were recorded for documentation purposes.

e) Transcription: Transcribing the audio-recordings resulted in research generated documents – the basis for the analyses. This was done using defined transcription rules following [67]. The transcripts cover 131 pages (M = 8.19; SD = 2.65) and 1,832 paragraphs (change of speaker) in total.

f) Coding Procedure: During the first step, the open coding, information is "broken down analytically" [68, p. 127]. The aim is to grasp the essence of the data, to find relevant information and capture it in codes – initially rather descriptively –, but also by using inductive reasoning to abstract and conceptualise. In contrast to other coding paradigms, where coding schemes are for example predefined, in GTM the codes emerge from the text. As coding and Constant Comparison evolve, codes, concepts, and memos can be abstracted and conceptualised on and on through axial and selective coding. This also comprises finding, arranging, and relating concepts and key categories (see Coding Paradigm [62]).

Through this process, Misconceptions of SE could be concluded through the use of GTM as an interpretative data analysis from the statements made by the undergraduates in the interviews using a Computer-Assisted Qualitative Data Analysis Software (in this case: MAXQDA [69]).

g) Investigator Triangulation: Multiple observers or interviewers are used to detect or minimize biases by the researcher. This can be done collaboratively or independently. How this is done in research practice and documented in publications has been inspected in a literature review by Archibald [70]. This revealed that out of 186 articles only 166 use the words "investigator" and "triangulation" incidentally in one sentence, but do not explain its usage [70].

In the research described in this paper, three researchers act as investigators, who are all research associates from the field of engineering and informatics[1] in order to be able to identify Misconceptions in SE; the disciplinary background is of prime importance (ref. Theoretical Sensitivity in Section V-A2). Two of them already conducted qualitative and quantitative research. Through the analysis and interpretation a consensus procedure for findings is applied. This is done in order to minimize the (constructivist [58, 11]) bias of a single coder and objectify[2] the results as well as maximize the richness of data and the reliability.

B. Construction of Misconception Items for the Catalogue

As seen elsewhere, concerning disparate [71], [72] as well as SE related disciplines [19], [22], [73] (and likewise: [20], [24]), Misconceptions are often formulated and documented as clauses of statement, e.g.:

- Physics: Driving a car at high speed consumes less fuel as it takes less time to drive [71, p. 17].
- Biology: Growing means to expand/enlarge [72, p. 371].
- Programming: A variable can hold multiple values at a time [19, p. 153].

Therefore, the concepts found encompassing Misconceptions in SE have been used to phrase declarative sentences as seen above (cf. Tables IX and X, 2nd column each). Care was taken to ensure that the formulations are as close as possible to the statements made by the undergraduates in the interviews (i.e., grounded in the data).

VI. Results

First it has to be noted that the catalogue was translated from German, which means additional transfer and therefore room for interpretation, although that is precisely why care was taken to stay as close as possible to the German formulations. However, one has to be aware that this is not only a result of the translation work, but also insists on a single language. This means that every person individually interprets every single word. From a constructivist perspective, everyone construes every word individually against the background of one's own knowledge, experiences, volition, motivation, and interest slightly differently from one another. Ergo, no identical or congruent statement will arrive at every single individual, regardless of which wording is used.

Tables IX and X (see next two pages), presenting the 69 items that have been reconstructed from the interviews with undergraduates (cf. Section V-A) and conclusively formulated accordingly (cf. Section V-B).

VII. Discussion

A. Implementation of GTM

Due to constructivism as stated before, not only undergraduates or pupils might hold Misconceptions concerning various topics and disciplines, but also researchers, teachers, lecturers, and professors. As there is – to the best of our knowledge – no possibility to avoid this, the approach of investigator triangulation was chosen in order to minimize the (constructivist) bias of a single coder and objectify[3] the results as well as maximize the richness of data and the reliability.

The established catalogue makes no claim to completeness, as also the interview guide developed and used is non-exhaustive. It should be noted that there will never be a complete list of misconceptions, since on the one hand there will always be something overlooked, which is human and impossible due to constructivism, and on the other hand – also following the constructivist paradigm – the conceptions of a single person is so extremely individual that a qualitative explorative investigation of "every" individual would be required. Given the fact that research on conceptions in Software Engineering has so far represented a research gap (cf. Sections II-IV; [1]), this study should embody the starting point on the path the natural sciences have already taken over the past decades in Discipline-Based Educational Research.

Additionally, the interviews have been conducted before a module in SE to get to know undergraduates (mis-)conceptions

[2]Of course, the one objective truth or reality can never be established, but in using multiple investigators/perspectives, an alternative to validation can be pursued.

[3]Of course, the one objective truth or reality can never be established, but in using multiple investigators/perspectives, an alternative to validation can be pursued.

TABLE IX
CATALOGUE OF UNDERGRADUATES' MISCONCEPTIONS ABOUT SOFTWARE ENGINEERING (PART I)

#	Items	Reference(s) for Comparison
A	**Software Development Process**	
A1	The phase of conception of a software project consists of organisation and project planning.	
A2	At the start of a software project, a basic source code is needed as a foundation.	
A3	The development of software is made up of the problem definition and its programming.	[49, p. 33; items 6 & 12]
A4	Software Engineering consists of programming and testing the software on its target hardware.	[49, p. 33; items 6 & 12]
A5	A FMEA (Failure Mode and Effects Analysis) is done after implementation.	
A6	Iterations and Optimization are seldom part of the Software Engineering process.	[53, p. 32; all items]
A7	The design phase in a software project is completed quickly.	
A8	Operation and Maintenance are not part of Software Engineering.	
A9	The process of software development is generally linear and sequential.	[44, p. 11; item 4L], [53, p. 32; all items]
A10	Both the development idea and the exact procedure for creating software are specified externally.	
B	**Requirements**	
B1	The customer requirements are fully present for the Software Engineer at the start of the project.	[49, p. 33; item 5]
B2	The work of the people employed in Software Engineering begins only when the specific task is available in detail.	[49, p. 33; item 5]
B3	The software project starts only when the complete assignment is available.	[49, p. 33; item 5]
B4	Usually customers know their requirements precisely.	[49, p. 33; item 5]
B5	As a rule, the requirements of business customers are unambiguous.	[49, p. 33; item 5]
B6	The customer's wishes are incorporated into a new product, but the basic product idea comes from within.	
B7	The requirements elicitation is the quickest part of a project.	
B8	Software normally only has to be implemented according to the conceptions of the Software Engineer.	
B9	The Programmer decides about additional functionalities of the software at his/her own discretion.	
C	**Implementation**	
C1	In a software project the implementation is the longest phase.	[44, p. 11; item 4B], [49, p. 33; items 6 & 12]
C2	A Software Engineer programmes on his/her own.	[49, p. 33; item 2]
C3	The implementation of a software project starts with the programming.	
C4	Software development is generally aimed at a specific hardware.	
D	**Testing and Bug Fixing**	
D1	The test phase can be concluded when no more bugs are found.	[44, p. 11; item 4M], [49, p. 33; item 9]
D2	In Software Engineering the work is successfully completed when there are no more defects, according to the customer's assessment.	
D3	The Software Engineer's job is successfully done when there are no more errors in the source code.	[44, p. 11; item 4M], [49, p. 33; item 9]
D4	Good software means everything works.	
D5	The source code of good software has to be free of errors.	[44, p. 11; item 4M], [49, p. 33; item 9]
D6	A Software Engineer does not conduct software tests.	
D7	Software is tested by trial and error.	
D8	A Software tester corrects the found errors him-/herself.	[47, p. 229; item 2], [49, p. 33; item 6]
E	**Scope of Delivery**	
E1	The source code is in general part of the scope of delivery.	
E2	Software development is completed when the artifacts are handed over to the customer.	

TABLE X
CATALOGUE OF UNDERGRADUATES' MISCONCEPTIONS ABOUT SOFTWARE ENGINEERING (PART II)

#	Items	Reference(s) for Comparison
F	**Comparison: Small – Large Software Development Companies**	
F1	Small software companies normally have less experience than big companies.	
F2	As a rule, small software companies work faster than big companies.	
F3	Big software companies work faster than small ones.	
F4	In big software companies software is optimized faster than in small companies.	
F5	In comparison to small software companies, employees in big companies are more skilled.	
F6	Large software companies invest more work into a software product.	
F7	Software developed by small companies does not have to run as stable as software from large companies.	
F8	In big software development companies more time is spent on testing a software.	
F9	Primarily large software development companies conduct market research.	
F10	Large companies generally distribute mass-produced products.	
F11	Software product development in large companies encompasses more steps than in small companies.	
G	**Comparison: Mass Products – Individual Commissioned Work**	
G1	When developing a mass product, no requirements elicitation is carried out, but the acceptance rate is evaluated retrospectively.	
G2	A commissioned work does not have to be as stable as a mass-produced product.	
G3	A mass product offers more functionalities than a commissioned work.	
G4	When developing a mass product, it has to be designed more intuitively than a commissioned work.	
G5	Mass products are generally developed independent of an operating system.	
G6	Mass products are typically more configurable than product from individual contracts.	
G7	When developing a mass product, what the software developers themselves consider appropriate is implemented.	
G8	The GUI of a mass-produced product is designed differently compared to that of commissioned work.	
G9	When developing a commissioned work, it is intended for a person with in-depth knowledge of the product.	
H	**Comparison: New Development – Extending Existing Software**	
H1	Developing a new software is more work intensive than to extend an existing one.	
H2	When developing a new software, you can start the programming immediately.	
H3	An extension is more complex if the basis is not an in-house development.	
H4	When software is well documented, it is easier to expand than to begin a new development.	
H5	When software is expanded its intended application cannot be changed anymore.	
H6	Today everything is an extension. Nothing is developed from scratch.	
I	**Working Life of a Software Engineer**	
I1	The main task of a Software Engineer is programming.	[43, p. 4; item A], [44, p. 11; item 4B], [49, p. 33; items 6 & 12]
I2	The job description of a Software Engineer does not differ from that of a computer scientist.	[43, p. 4; item A]
J	**Customer Contact**	
J1	Customer contact is normally not performed by Software Engineers.	[43, p. 4; item D], [49, p. 33; item 11]
J2	In larger companies customer contact is in general not undertaken by Software Engineers.	[43, p. 4; item D], [49, p. 33; item 11]
J3	Customer dialogue is not part of Software Engineering itself.	[43, p. 4; item D], [49, p. 33; item 11]
K	**Overview**	
K1	An overview of the complete software development process is not required for working as a Software Engineer.	[49, p. 33; items 8 & 11]
K2	For working as a Software Engineer, it is sufficient to be familiar with your assigned task.	[49, p. 33; items 8 & 11]
K3	A Software Engineer does not need to have an overview of the finances of the project.	[49, p. 33; items 8 & 11]
L	**Miscellaneous**	
L1	Compared to nontechnical users, engineers can handle software even if it is not designed intuitively.	
L2	Programmers normally only have contact among each other.	

as they enter. Therefore, the interviews undertaken covered questions on a relatively coarse-granular level (see Sec. V-A).

B. Validaion of Results

The Misconceptions found have also been validated by 13 experts, which is explained and preseted in detail in the publication [2]. This is done for mainly two reasons:

- First, all individuals are subject to constructivism, so researchers and lecturers have to be aware that not only students/undergraduates/learners, but also researchers and lecturers might have Misconceptions – even concerning our own field of expertise (see "Investigator Triangulation" in Sec. V-A2) as stated by Duit [58, p. 11], who is himself a professor emeritus for the didactics of physics.
- Furthermore, the line between "technically adequate" and "inadequate" can be drawn relatively clearly in relation to Misconceptions in the natural sciences, since these are based on natural laws and rules. To a large extent, this can also be transferred to Computer Sciences fields, such as programming, since language definitions also specify "correct"/"incorrect" (e.g., syntactically speaking).
  In Software Engineering, on the other hand, this line does not seem to be so clear and easy to draw [74, p. 8 f.] [75, p. 12].
  First, SE does not follow any nomothetic premises, unlike the natural sciences, which aim to transfer their knowledge into regularities and thus try to explain the world through natural laws and rules – often mathematically [74, p. 8 f.] [75, p. 12]. There are simply no formulas or similar, as in the natural sciences and mathematics, which when applied lead to proven and guaranteed correct results [75, p. 12].
  Second, neither in SE nor in many other areas of computer science (e.g., programming), *one correct* (sample[4]) solution exists, but a whole solution space [76, p. 437], which is available. On the one hand, this is due to the human factor, which, for example, has a decisive influence on the result of a (large) software development project. On the other hand, there is also the intrinsic plurality of possible solutions that result from the implementation alternatives, so that subsequently in SE a solution path will probably never be reproduced identically twice [75, p. 12 ff.].

VIII. Analysis and Comparison

To combine the two pillars explained in detail beforehand, this third part is intended to analyse and compare the findings of both elicitations. It has to be noted that the aim is not to conduct a one-to-one matching, but to look on context-related proximities, to search for same or similar underlying ideas and thinking patterns (see Section VIII-A) as well as – partly based upon this – have a look at the thinking patterns and make considerations concerning a root cause analysis on MCs in SE (see Section VIII-B).

[4]"Sample solution" based on the didactic teaching context.

A. Comparison and Contrasting

Besides the MCs found in the interviews, Tables IX and X also encompass a third column covering references of MCs found through the SLR, which have a similar meaning or same underlying concept. For example, B2 (Table IX: *The work of the people employed in Software Engineering generally begins only when the specific task is available in detail.* is relatively similar to "It is expected that clients will describe their requirements accurately before a team begins programming." [49, p. 33; item 5]. The comparison of the survey results and the SLR results has been done on a meta level, since neither the same wording nor the exactly identical meaning is found when contrasting the item lists. This compasion has been done on the basis of the 69 items constructed from the MCs found through the interview study on the one site and the MCs identified in the SLR before the application of the quality criteria on the other site. This has been done, since the quality criteria have been established and used to filter for evidence-based MCs, which does not mean that these might be MCs although they do not meet the criteria.

The comparison shows several overlaps – especially concerning some items that also have overlaps within the SLR; e.g., A9, C1, I1, J1-3.

Finally, it should be noted: The fact that several items in Tables IX and X have not been "found" in the SLR and vice versa does not imply that the interviewees do not hold them. This may be due to the various reasons:

- Oral articulateness possibilities (cf. discipline-specific language and terminology)
- Open ended questions in die interviews
- Undergraduates are new to SE
- Qualitiative interviews assert no claim for completeness of results

B. Further Analyses

What can also be seen is that even larger clusters/categories can be formed through further abstraction; supplemented by many MC candidates from the SLR. One very large topic are complexity reductions such as (excerpt):

- One-dimensional (restricted) programmer perspective: e.g., A3, A4, A7, A8, C1, C3, [43, p. 4; item A], [44, p. 11; item 4B], [49, p. 33; items 6, 8 & 12]
- Linearity and sequentiality of the process: e.g., A6, A9, [44, p. 11; item 4L], [53, p. 32; all items]
- Simplification of requirements elicitation: e.g., B1-B5, B7, [49, p. 33; items 5],
- Focusing on a sub-area of the SE: e.g., K1-K3, [49, p. 33; items 8],

These thinking patterns could also be interpreted as possible causes for MCs.

IX. Summary & Outlook

The paper's purpose, to identify and analyse misconceptions in SE to use these findings in higher education, has been pursued using a systematic literature review for already known MCs as well as a interview study getting new insights.

Looking at the SLR, predefined search queries have been applied to search 10 databases before the publications have been filtered using the selection strategy described. Out of 2,158 publications, 18 could be identified as appropriate for the selection criteria. These contain 167 statements, which the authors of these papers refer to as misconceptions. 20 of them met the quality criteria specified; i.e., only 3 publications cover valuable data.

To conclude, the results show that currently evidence-based research on misconceptions in SE is limited. So, in addition a primary study to identify misconception in SE is indispensable before addressing them.

Therefore, the procedure of qualitative data elicitation, analysis, and interpretation has been exploited to detect Misconceptions in SE through Straussian's Grounded Theory Methodology [62]. Building on these findings, a brief look was taken at the subsequent formulation of misconceptions in accordance with previous literature (see Sec. V-B).

Bringing the results of both approaches and data sources together, a comparison on a meta level has been conducted. As a result, it can be stated that there are similarities concerning thinking patterns, which have to be further investigated.

As an outlook, in order to strive to achieve sustainable learning (in higher education) – as displayed in the introduction – another step is taken to get an understanding of undergraduates conceptions in order to do justice to the learners' "points of departure" [3, p. 6] and misconcetpions as learning obstacles [10].

Acknowledgement

The present publication is based on the work of the EVELIN project funded by the German Federal Ministry of Education and Research (BMBF) under grant number 01PL17022B. The authors are responsible for the content of this publication.
The author thanks the reviewers of the ICSEA 2020 conference and the IARIA Journal to have been selected for the invitation to submit this paper; beyond the very useful comments that have contributed to enhance both the original and the extended version of this paper.
Above all, I would like to thank my two co-investigators during the undergraduate survey [2], Rebecca Reuter and Sabrina Jahn, both from the OTH Regensburg, Germany as well as Theresa Stark for her support.
Also, I would like to thank Nermin Saray for her support in conducting the SLR [1].

References

[1] C. Gold-Veerkamp and N. Saray, "A Systematic Literature Review on Misconceptions in Software Engineering," in *Proceedings of the Fifteenth International Conference on Software Engineering Advances (ICSEA)*, 2020, pp. 1–8.

[2] C. Gold-Veerkamp, "Validated Undergraduates' Misconceptions about Software Engineering," in *Proceedings of the 2021 IEEE Global Engineering Education Conference (EDUCON)*, Institute of Electrical and Electronics Engineers, Ed. Piscataway, NJ: IEEE, 2021, pp. 609–618.

[3] R. Duit, "Science education research internationally: Conceptions, research methods, domains of research," *EURASIA Journal of Mathematics, Science & Technology Education*, vol. 3, no. 1, pp. 3–15, 2007.

[4] R. Duit, H. Gropengießer, U. Kattmann, M. Komorek, and I. Parchmann, "The model of educational reconstruction," in *Science Education Research and Practice*, D. Jorde and J. Dillon, Eds. Springer, 2012, pp. 13–37.

[5] J. P. Smith, A. A. diSessa, and J. Roschelle, "Misconceptions reconceived: A constructivist analysis of knowledge in transition," *Journal of the Learning Sciences*, vol. 3, no. 2, pp. 115–163, 1994.

[6] J. R. Read, "Children's misconceptions and conceptual change in science education," 2004, [retrieved: 09, 2020]. [Online]. Available: http://acell.chem.usyd.edu.au/Conceptual-Change.cfm

[7] I. Diethelm and S. Zumbrägel, "An investigation of secondary school students' conceptions on how the internet works," in *Koli Calling International Conference on Computing Education Research*, M.-J. Laakso and R. McCartney, Eds. ACM Press, 2012, pp. 67–73.

[8] S. Todtenhaupt, *To develop an understanding of chemistry in schoolchildren: An investigation into the redox topic at a high school. (Original title: "Zur Entwicklung des Chemieverständnisses bei Schülern: Eine Untersuchung zur Redox-Thematik an einem Gymnasium" [German])*. Frankfurt a.M.: Lang, 1995.

[9] A. Taylor Kujawski and P. Kowalski, "Naïve psychological science: the prevalence, strength, and source of misconceptions," *The Psychological Record*, vol. 54, pp. 15–25, 2004.

[10] R. Reuter, F. Hauser, C. Gold-Veerkamp, J. Mottok, and J. Abke, "Towards a definition and identification of learning obstacles in higher software engineering education," in *Annual International Conference on Education and New Learning Technologies (EDULEARN)*, IATED, Ed., 2017, pp. 10 259–10 267.

[11] R. Duit, "STCSE: Students' and teachers' conceptions and science education," Bibiography, 2009, [retrieved: 09, 2020]. [Online]. Available: http://archiv.ipn.uni-kiel.de/stcse/download_stcse.html

[12] D. N. Perkins and R. Simmons, "Patterns of misunderstanding: An integrative model of misconceptions in science, math and programming," in *2. Int. Seminar Misconceptions and Educational Strategies in Science and Mathematics: Vol. I*, J. D. Novak, Ed., 1987, pp. 381–395.

[13] ——, "Patterns of misunderstanding: An integrative model for science, math, and programming," *Review of Educational Research*, vol. 58, no. 3, pp. 303–326, 1988.

[14] L. Louca and Z. C. Zacharia, "The use of computer-based programming environments as computer modelling tools in early science education: The cases of textual and graphical program languages," *International Journal of Science Education*, vol. 30, no. 3, pp. 287–324, 2008.

[15] J. Confrey, "Misconceptions across subject matter: science, mathematics, programming," in *2. Int. Seminar Misconceptions and Educational Strategies in Science and Mathematics: Vol. I*, J. D. Novak, Ed., 1987, pp. 81–106.

[16] N. Taylor and G. Corrigan, "New South Wales primary school teachers' perceptions of the role of ICT in the primary science curriculum - A rural and regional perspective," *International Journal of Science and Mathematics Education*, vol. 5, no. 1, pp. 85–109, 2007.

[17] R. D. Pea, "Language-independent conceptual "bugs" in novice programming," *Journal educational computing research*, vol. 2, no. 1, pp. 25–36, 1986.

[18] J. Sorva, "Visual program simulation in introductory programming education." Dissertation, Espoo, Aalto Univ., Finland, 2012.

[19] A. Swidan, F. Hermans, and M. Smit, "Programming misconceptions for school students," in *Conference on International Computing Education Research (ICER)*. ACM, 2018, pp. 151–159.

[20] Ž. Žanko, M. Mladenović, and I. Boljat, "Misconceptions about variables at the K-12 level," *Education and Information Technologies*, vol. 24, no. 2, pp. 1251–1268, 2019.

[21] R. Kelter, M. Kramer, and T. Brinda, "Statistical frequency-analysis of misconceptions in object-oriented-programming: Regularized pcr models for frequency analysis across oop concepts and related factors," in *Koli Calling International Conference on Computing Education Research*, M. Joy and P. Ihantola, Eds. ACM, 2018, pp. 6:1–6:10.

[22] S. Holland, R. Griffiths, and M. Woodman, "Avoiding object misconceptions," in *SIGCSE technical symposium on Computer science education*, C. M. White, C. Erickson, B. Klein, and J. E. Miller, Eds. ACM, 1997, pp. 131–134.

[23] M. T. Rücker and N. Pinkwart, ""How else should it work?" A grounded theory of pre-college students' understanding of computing devices," *ACM Transactions on Computing Education*, vol. 19, no. 1, pp. 2:1–2:23, 2018.

[24] I. Diethelm, P. Hubwieser, and R. Klaus, "Students, teachers and phenomena: Educational reconstruction for computer science education," in *Koli Calling International Conference on Computing Education Research*, M.-J. Laakso and R. McCartney, Eds. ACM, 2012, pp. 164–173.

[25] B. A. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering: Version 2.3," EBSE Technical Report (EBSE-2007-01), Keele University & University of Durham, 2007.

[26] P. O. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain," *Journal of Systems and Software*, vol. 80, no. 4, pp. 571–583, 2007.

[27] B. A. Kitchenham et al., "Systematic literature reviews in software engineering – a systematic literature review," *Information and Software Technology*, vol. 51, no. 1, pp. 7–15, 2009.

[28] A. Bartel, "Conception and development of a DSM-based gamification authoring system to support university teaching. (Original title: "Konzeption und Entwicklung eines DSM-basierten Gamification Authoring Systems zur Unterstützung hochschulischer Lehre" [German])," Dissertation, Universität Regensburg, 2018.

[29] IEEE, "IEEE Xplore Digital Library," 2020. [Online]. Available: https://ieeexplore.ieee.org/Xplore/home.jsp

[30] ACM, "ACM Digital Library," 2020. [Online]. Available: http://dl.acm.org/

[31] Springer Nature Switzerland AG, "SpringerLink," 2020. [Online]. Available: https://link.springer.com/

[32] Elsevier B.V., "Scopus," 2020. [Online]. Available: http://www.scopus.com/

[33] ——, "ScienceDirect," 2020. [Online]. Available: https://www.sciencedirect.com/

[34] Institute of Education Sciences of the US Department of Education, "Eric – education resources information center," 2020. [Online]. Available: https://eric.ed.gov/

[35] Clarivate Analytics, "Web of Science," 2020. [Online]. Available: http://www.webofknowledge.com

[36] Google LLC, "Google Scholar," 2020. [Online]. Available: http://scholar.google.de/

[37] Cornell University, "arXiv.org: e-Print archive," 2020. [Online]. Available: https://arxiv.org/

[38] Schloss Dagstuhl and Universität Trier, "dblp: computer science bibliography," 2020. [Online]. Available: https://dblp.uni-trier.de/

[39] B. A. Kitchenham and P. O. Brereton, "A systematic review of systematic review process research in software engineering," *Information and Software Technology*, vol. 55, no. 12, pp. 2049–2075, 2013.

[40] Z. Wan et al., "Perceptions, expectations, and challenges in defect prediction," *IEEE Transactions on Software Engineering*, pp. 1–26, 2018.

[41] N. C. Shrikanth and T. Menzies, "Assessing practitioner beliefs about software defect prediction," 2020, accepted at ICSE'20, [retrieved: 09, 2020]. [Online]. Available: arXiv:1912.10093v3

[42] P. Devanbu, T. Zimmermann, and C. Bird, "Belief evidence in empirical software engineering," in *International Conference on Software Engineering (ICSE)*, 2016, pp. 108–119.

[43] M. M. Inuwa and A. Varol, "Intensity of misconception in software engineering," in *International Informatics and Software Engineering Conference (UBMYK)*, 2019, pp. 1–6.

[44] J. Ivins, B. R. von Konsky, D. Cooper, and M. Robey, "Software engineers and engineering: A survey of undergraduate preconceptions," in *Frontiers in Education (FIE)*, 2006, pp. MIF–6–11.

[45] B. Özkan and O. Demirors, "On the seven misconceptions about functional size measurement," in *Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA)*, 2016, pp. 45–52.

[46] J. S. van der Ven and J. Bosch, "Busting software architecture beliefs: A survey on success factors in architecture decision making," in *Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, 2016, pp. 42–49.

[47] A. Begel and B. Simon, "Struggles of new college graduates in their first software development job," *SIGCSE Bull*, vol. 40, no. 1, pp. 226–230, 2008.

[48] D. DeMarco Brown, "Five agile ux myths," *Journal of Usability Studies*, vol. 8, no. 3, pp. 55–60, 2013.

[49] L. A. Sudol and C. Jaspan, "Analyzing the strength of undergraduate misconceptions about software engineering," in *International Computing Education Research (ICER)*. ACM, 2010, pp. 31–39.

[50] R. H. Wilcox, "Behavioral misconceptions facing the software engineer," in *Computer and Information Sciences*, ser. SEN Report Series Software Engineering, J. T. Tou, Ed. Elsevier, 1971, vol. 2, pp. 285–287.

[51] J. P. Bowen and M. G. Hinchey, "Seven more myths of formal methods: Dispelling industrial prejudices," in *FME'94: Industrial Benefit of Formal Methods*, ser. LNCS 873. Springer, 1994, pp. 105–117.

[52] A. Hall, "Seven myths of formal methods," *IEEE Software*, vol. 7, no. 5, pp. 11–19, 1990.

[53] D. Carlson, "Debunking agile myths," *CrossTalk*, vol. 30, no. 3, pp. 32–36, 2017.

[54] S. Jahn, C. Gold-Veerkamp, R. Reuter, J. Mottok, and J. Abke, "Secure software engineering in academic education: Students' preconceptions of it security," in *International Conference of Education, Research and Innovation (ICERI)*, IATED, Ed., 2019, pp. 6825–6834.

[55] P. Ralph and B. J. Oates, "The dangerous dogmas of software engineering," 2018, [retrieved: 09, 2020]. [Online]. Available: arXiv:1802.06321v1

[56] N. C. Shrikanth and T. Menzies, "Assessing developer beliefs: A reply to "perceptions, expectations, and challenges in defect prediction"," 2019, [retrieved: 09, 2020]. [Online]. Available: arXiv:1904.05794v1

[57] D. Rombach and F. Seelisch, "Formalism in software engineerings: Myths versus empirical facts," in *Balancing Agility and Formalism in Software Engineering*, D. Hutchison, et al., Ed. Springer, 2008, pp. 13–25.

[58] R. Duit, "Schülervorstellungen – von Lerndefiziten zu neuen Unterrichtsansätzen [German]," in *Schülervorstellungen in der Physik*, R. Müller, R. Wodzinski, and M. Hopf, Eds. Köln: Aulis, 2011, pp. 8–14.

[59] C. Schulte and J. Magenheim, "Novices' expectations and prior knowledge of software development," in *First international Workshop on Computing education research*, R. Anderson, S. A. Fincher, and M. Guzdial, Eds. ACM Press, 2005, pp. 143–153.

[60] C. Gold-Veerkamp, "Big picture: für das evelin-doktorandenkoll." 2018.

[61] B. G. Glaser, *Basics of Grounded Theory Analysis – Emergence vs. Forcing*, 1st ed. Mill Valley, CA: Sociology Press, 1992.

[62] J. M. Corbin and A. L. Strauss, *Basics of qualitative research: Techniques and procedures for developing grounded theory*, fourth edition ed. Los Angeles and London and New Delhi and Singapore and Washington DC and Boston: SAGE Publications Ltd, 2015.

[63] K. Charmaz, *Constructing Grounded Theory*, 2nd ed. London: SAGE Publications Ltd, 2014.

[64] K.-J. Stol, P. Ralph, and B. Fitzgerald, "Grounded theory in software engineering research: A critical review and guidelines," in *The 38th International Conference on Software Engineering (ICSE)*, L. Dillon, W. Visser, and L. Williams, Eds. ACM Press, 2016, pp. 120–131.

[65] C. Gold-Veerkamp, J. Abke, and I. Diethelm, "What about misconceptions in software engineering? a research proposal," in *Proceedings of the 2017 IEEE Global Engineering Education Conference (EDUCON)*, Institute of Electrical and Electronics Engineers, Ed. Piscataway, N.J.: IEEE, 2017, pp. 709–713.

[66] B. G. Glaser, *Theoretical sensitivity: Advances in the methodology of grounded theory*. Mill Valley, Calif.: Sociology Press, 1978.

[67] M. Selting, P. Auer, D. Barth-Weingarten, J. Bergmann, P. Bergmann, K. Birkner, E. Couper-Kuhlen, Arnulf Deppermann, P. Gilles, S. Günthner, M. Hartung, F. Kern, C. Mertzlufft, C. Meyer, M. Morek, F. Oberzaucher, J. Peters, U. Quasthoff, W. Schütte, Anja Stukenbrock, and S. Uhmann, "Gesprächsanalytisches Transkriptionssystem 2 (GAT 2)," *Gesprächsforschung - Online-Zeitschrift zur verbalen Interaktion*, no. Ausgabe 10, pp. 353–402, 2009.

[68] A. Boehm, "Grounded theory: Wie aus texten modelle und theorien gemacht werden," in *Texte verstehen*, A. Boehm, A. Mengel, and T. Muhr, Eds. Konstanz: UVK Verlagsgesellschaft mbH, 1994, pp. 121–140.

[69] VERBI Software GmbH, "Maxqda: All-in-one qualitative & mixed methods data analysis tool," 2020, (last access 2021-02-28). [Online]. Available: https://www.maxqda.com/

[70] M. M. Archibald, "Investigator triangulation: A collaborative strategy with potential for mixed methods research," *Journal of Mixed Methods Research*, vol. 10, no. 3, pp. 228–250, 2016.

[71] B. Eisner, U. Kattmann, M. Kremer, J. Langlet, D. Plappert, and B. Ralle, "Gemeinsamer referenzrahmen für naturwissenschaften

(gerrn): Mindeststandards für die auf naturwissenschaften bezogene bildung. ein vorschlag."

[72] U. Kattmann, *Schüler besser verstehen: Alltagsvorstellungen im Biologieunterricht : [zusätzliche Stichwörter zum Download*. Hallbergmoos: Aulis Verlag, 2015.

[73] M. T. Rücker and N. Pinkwart, "Review and discussion of children's conceptions of computers," *Journal of Science Education and Technology*, vol. 25, no. 2, pp. 274–283, 2016.

[74] J. Erpenbeck, Ed., *Der Königsweg zur Kompetenz: Grundlagen qualitativ-quantitativer Kompetenzerfassung*, ser. Kompetenzmanagement in der Praxis. Münster and New York and München and Berlin: Waxmann, 2012, vol. Band 6.

[75] Y. Sedelmaier, "Interdisziplinäre Fachdidaktik für Software Engineering: Forschungsbasierte Entwicklung und Evaluation eines anwendungsbezogenen didaktischen Ansatzes," Ph.D. dissertation, Otto-Friedrich-Universität, Bamberg, 2015.

[76] H. Balzert, *Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements-Engineering*, 3rd ed., ser. Lehrbücher der Informatik. Heidelberg: Spektrum Akad. Verl., 2009.