Erste Schritte zu einer Taxonomie von Software Engineering Kontexten

*Eine **Masterarbeit** vorgelegt für den Abschluss **Master of Science**.*

Stephanie Hohenberg[1], Freie Universität Berlin, Germany

Matrikelnummer: 5293431

stephanie.hohenberg@fu-berlin.de

5. September 2021

Betreuer:

Prof. Dr. Lutz Prechelt[2], Freie Universität Berlin, Germany

Prof. Dr. Claudia Müller-Birn[3], Freie Universität Berlin, Germany

Zitation:

Stephanie Hohenberg, *Erste Schritte zu einer Taxonomie von Software Engineering Kontexten*, Freie Universität Berlin, Masterarbeit, 2021

[1]Fachbereich Mathematik und Informatik, Institut für Informatik
[2]Institut für Informatik, Software Engineering Arbeitsgruppe
[3]Institut für Informatik, Human-Centered Computing Arbeitsgruppe

Selbstständigkeitserklärung

Ich erkläre gegenüber der Freien Universität Berlin, dass ich die vorliegende Masterarbeit selbstständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe. Die vorliegende Arbeit ist frei von Plagiaten. Alle Ausführungen, die wörtlich oder inhaltlich aus anderen Schriften entnommen sind, habe ich als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch bei keiner anderen Universität als Prüfungsleistung eingereicht.

Berlin, 5. September 2021

Stephanie Hohenberg

Abstract

Background: The *International Conference on Software Engineering* (ICSE) is one of the leading conferences for software engineering. The papers accepted for that conference pertain to various contexts, which span over various dimensions and relate to each other. These contexts can be mapped in a taxonomy.

Objectives: In this master thesis first steps towards such a taxonomy describing software engineering contexts, their dimensions and their relations will be done. The objective is to create a terminology and to improve the communication in the software engineering world. This taxonomy has the potential to support different user groups in locating the current research.

Methods: To reach these objectives, a literature study was conducted, reviewing the technical papers of the ICSE 2020 to identify the contexts of these papers and to shape a taxonomy.
Furthermore a prototype, called *TSEK-Analyzer*, was implemented, to visualize this taxonomy, to offer an overview of the results of this thesis and to the data analysis within this master thesis.
To evaluate the resulting taxonomy, the authors of the reviewed papers were asked to validate the classification of their work in the taxonomy. Furthermore semi-structured interviews with members of the ICSE program board 2020 were held and usability tests with software engineers for the prototype were conducted to evaluate the relevance of this thesis and to gain insights of potential user groups.

Results: A taxonomy with the following dimensions was created: *software engineering activities, software tools, applications, data* and *human aspects*, covering 59 of the 129 technical papers of the ICSE 2020. Moreover, use case scenarios specific for different potential user groups got analysed and out of them the next steps to integrate a taxonomy of software engineering contexts into the market got mapped out.

Abstract

Hintergrund: Die Internationale Konferenz für Software Engineering (ICSE) ist eine der größten Konferenzen, um aktuelle akademische Beiträge der Software Engineering Welt vorzustellen. Die Arbeiten, die auf dieser Konferenz präsentiert werden, bewegen sich in verschiedenen Kontexten, welche sich über unterschiedlichen Dimensionen spannen und in Beziehungen zwischen einander stehen. Diese Kontexte können in einer Taxonomie abgebildet werden.

Ziele: In dieser Masterarbeit werden erste Schritte zu einer solchen Taxonomie getätigt, welche Software Engineering Kontexte, dessen Dimensionen und Beziehungen zwischen diesen beschreibt. Ziel dabei ist es eine Terminologie zu schaffen und die Kommunikation in der Software Engineering Welt zu verbessern.
Ein solche Taxonomie kann verschiedene Personengruppen bei der Verortung verschiedener Forschungsarbeiten unterstützen.

Methoden: Um diese Ziele zu erreichen, wurde in dieser Masterarbeit eine Literaturstudie über die technischen Arbeiten der ICSE 2020 durchgeführt, um die Kontexte dieser Arbeiten zu identifizieren und die Verallgemeinerbarkeit ihrer Ergebnisse zu analysieren.
Des Weiteren wurde ein Prototyp entwickelt, um die entstehende Taxonomie aus den identifizierten Kontexten zu visualisieren, eine Übersicht über die Ergebnisse dieser Masterarbeit zu bieten und die Datenanalyse innerhalb dieser Masterarbeit zu unterstützen.
Um die Taxonomie zu evaluieren, wurden die Autorinnen und Autoren der untersuchten Paper gebeten an einer Umfrage teilzunehmen. Außerdem wurden Usability Tests mit Software Ingenieurinnen und Software Ingenieuren für den Prototypen *TSEK-Analyzer* durchgeführt und semi-strukturierte Interviews mit Gutachterinnen und Gutachtern, die Mitgliederinnen und Mitglieder des Programm Vorstands der ICSE sind, gehalten, um die Relevanz dieser Masterarbeit zu evaluieren und um die potenziellen Nutzergruppen kennen zu lernen,

Ergebnisse: Eine Taxonomie mit den folgenden Dimensionen wurde entwickelt: *software engineering activities, software tools, applications, data* und *human aspects*. Außerdem wurden Nutzungsszenarien, zugeschnitten auf bestimmte Personengruppen, analysiert. Mithilfe dieser wurden die nächsten Schritte für die Integration einer Taxonomie von Software Engineering Kontexten auf den Markt herausgearbeitet.

Inhaltsverzeichnis

1 Einführung **7**
1.1 Motivation . . . 7
1.2 Struktur der Masterarbeit . . . 8

2 Hintergrund **9**
2.1 Kontextbeschreibung von empirischen Arbeiten . . . 9
2.2 International Conference on Software Engineering . . . 9
2.3 Verwandte Arbeiten . . . 11

3 Vorgehen **13**
3.1 Sichtung der Arbeiten . . . 13
3.1.1 Identifizierung von Kontexten . . . 15
3.1.2 Anfertigung einer Taxonomie . . . 17
3.1.3 Analyse zur Verallgemeinerbarkeit . . . 17
3.2 Dokumentation . . . 18

4 Durchführung **19**
4.1 Charakteristiken der Arbeiten . . . 19
4.2 Analyse zur Verallgemeinerbarkeit . . . 21
4.3 Analyse der Kontexte . . . 25
4.4 Anfertigung der Taxonomie . . . 28
4.5 Implementierung des Prototyps: TSEK-Analyzer . . . 29

5 Evaluation **33**
5.1 Evaluation des Vorgehen . . . 33
5.2 Evaluation der Relevanz . . . 35
5.2.1 Segmentierung . . . 36
5.2.2 Semi-Strukturierte Interviews mit Gutachterinnen und Gutachtern 38
5.2.3 Analyse weiterer Segmente . . . 43
5.3 Evaluation des Prototyps: TSEK-Analyzer . . . 45
5.3.1 SetUp . . . 45
5.3.2 Ergebnisse der Usability Tests . . . 48
5.3.3 Ergebnisse der semi-strukturierten Interviews . . . 50
5.4 Evaluation der Taxonomie . . . 53
5.4.1 Ergebnisse der Umfrage . . . 53
5.4.2 Vergleich der Taxonomie . . . 57

6 Abschluss **58**
6.1 Zusammenfassung der Ergebnisse . . . 58
6.2 Beantwortung der Forschungsfragen . . . 59
6.3 Diskussion zur Relevanz und Verallgemeinerbarkeit . . . 60
6.4 Nutzungsszenarien und Marktstrategie . . . 61
6.5 Ausblick . . . 62

7 Anhang **64**
7.1 Taxonomie 64
7.2 Übersicht über die gesichteten Arbeiten 70
7.3 Test- und Fragebogen zum Usability Test und zum semi-strukturierten Interview 83
7.4 Anschreiben und Fragebogen zu den semi-strukturierten Interviews mit den ICSE Program Board 84
7.5 Anschreiben und Umfrage an die Autorinnen und Autoren 87

8 Literatur **89**

1 Einführung

Im Folgenden wird die Motivation zum Verfassen dieser Masterarbeit erläutert. Außerdem wird dessen Struktur beschrieben.

1.1 Motivation

Software Engineering ist eine Disziplin in der Informatik, welche viele Themen und Richtungen für die Forschung bietet. Die Beweiserhebung in der Forschung ist dabei Kontext-spezifisch [39], sodass die Ergebnisse einer Forschungsarbeit innerhalb dessen Kontext betrachtet werden sollte.

Um besser verorten zu können, in welchen Kontexten sich die aktuelle Forschung im Bereich Software Engineering bewegt, sollen, wie der Titel dieser Masterarbeit bereits verrät, erste Schritte zur Entwicklung einer Taxonomie von Kontexten unternommen werden.

Eine solche Taxonomie kann nicht nur bei der Verortung der Forschung unterstützen, sondern auch eine Terminologie zur Beschreibung des Kontexts einer Arbeit schaffen, welche zur Verbesserung der Kommunikation führen kann und zwar nicht nur zwischen Forschenden, die empirische Arbeiten verfassen, sondern auch zwischen Gutachterinnen und Gutachtern, die dessen Arbeiten bewerten. Ebenfalls Studierenden, die empirische Arbeiten sichten und evaluieren und ihre ersten empirische Arbeiten verfassen kann eine solche Taxonomie die Kommunikation und Verortung von Forschungsarbeiten erleichtern, sowie Software Ingenieurinnen und Ingenieuren, die erstreben auf den aktuellsten Stand ihres Feldes zu sein.
Die Taxonomie kann die Personen dieser verschiedenen Gruppen leiten, sodass die Relevanz einer Arbeit für die eigenen Zwecke besser eingeschätzt werden kann und somit die Übertragbarkeit von Ergebnissen gefördert wird.

Zur Anfertigung einer derartigen Taxonomie wurde eine Literaturstudie durchgeführt um Kontexte in aktuellen Forschungsarbeiten zu identifizieren. 59 technische Arbeiten, die auf der *International Conference on Software Engineering* 2020, im Weiteren abgekürzt als ICSE, veröffentlicht wurden, wurden gesichtet und dessen Kontexte zu einer ersten Taxonomie zusammengeführt.

1.2 Struktur der Masterarbeit

Die Masterarbeit besteht aus sechs Kapitel, deren Inhalte und Struktur im Folgendem erläutert wird.

Zunächst wird in **Kapitel 2 *Hintergrund***, näher auf den Begriff des Kontexts eingegangen, sowie die Konferenz ICSE detailliert vorgestellt. Des Weiteren werden *verwandte Arbeiten* vorgestellt und erläutert.

In **Kapitel 3 *Vorgehen*** werden die Aktivitäten beschrieben, die innerhalb der Literaturstudie durchgeführt wurden. Dabei wird auch erläutert, inwiefern Werkzeuge zur Dokumentation verwendet und entwickelt wurden.

In dem darauf folgendem **Kapitel 4 *Durchführung*** wird die Realisierung des Vorgehens dokumentiert und die Ergebnisse zu den Aktivitäten der Literaturstudie werden erörtert. Ebenfalls wird die Implementierung eines Prototyps zur Visualisierung der Taxonomie beschrieben.

Anschließend wird in **Kapitel 5 *Evaluation*** das Vorgehen, sowie die Relevanz dieser Arbeit, die entstandene Taxonomie, und der Prototyp zur Visualisierung der Taxonomie evaluiert. Dazu wurden zusätzlich zu der Literaturstudie Interviews, Usability Tests und eine Umfrage durchgeführt.

Im letzten **Kapitel 6 *Abschluss*** werden die Ergebnisse und die Schlussfolgerungen dieser Arbeit zusammengefasst und über die Relevanz und Verallgemeinerbarkeit dieser Arbeit diskutiert. Außerdem wird ein Ausblick auf weitere Forschung und auf weitere Schritte in Form einer Marktstrategie gegeben.

2 Hintergrund

Im Folgendem wird der Hintergrund der Masterarbeit vorgestellt. Dazu wird eine Definition von dem Begriff Kontext gegeben, die *International Conference on Software Engineering* vorgestellt und verwandte Arbeiten werden präsentiert.

2.1 Kontextbeschreibung von empirischen Arbeiten

Innerhalb dieser Arbeit sollen die Kontexte von empirischen Arbeiten untersucht werden und zu einer Taxonomie zusammengefasst werden. Im Rahmen dieser Arbeit wird der Begriff Kontext dabei wie folgt definiert.

Definition: Ein Kontext beschreibt den Zusammenhang, in denen Ergebnisse einer Arbeit gelten oder übertragbar sind.

Dabei spielt unter Anderem das empirische Vorgehen und der Kontext, in welchem eine Arbeit Forschungsfragen untersucht, eine Rolle. Des Weiteren lässt die Beschreibung der Relevanz und der Verallgemeinerbarkeit von Ergebnissen auf einen größeren Kontext schließen, auf welchen die Ergebnisse übertragbar sind.
Es ist zu erwarten, dass Kontexte an verschiedenen Stellen in einer Arbeit Erwähnung finden und mit verschiedenen Absichten von den Autorinnen und Autoren beschrieben werden.

2.2 International Conference on Software Engineering

Die *International Conference on Software Engineering*[4], im Weiteren abgekürzt als ICSE, ist eine international anerkannte Konferenz für Software Engineering.

Die Zielgruppe der ICSE sind Forscherinnen und Forscher, Praktizierende und Lehrende in der Disziplin Software Engineering. Es wird ein Forum geboten um die aktuellsten Forschungsbeiträge zu diskutieren, die neuesten Innovationen und Trends zu präsentieren, sowie Erfahrungen und Probleme zu teilen.[5]

[4] https://conf.researchr.org/home/icse-2020
[5] https://2020.icse-conferences.org/

Im Jahr 2020 fand die Konferenz zum 42. Mal und zum erstem Mal virtuell statt. Es wurden 617 technische Paper von Autorinnen und Autoren aus der ganzen Welt eingereicht. Der Prozess vom Einreichen bis zum Akzeptanz einer Arbeit kann in sechs Phasen eingeteilt werden:

1. Einreichen der Arbeiten bis zur Frist des so genannten *Call for Papers*
2. Aussortieren von Arbeiten aufgrund von Verstößen gegen Formatierungsregeln, Seitenbeschränkungen oder Plagiat
3. Begutachtung der restlichen Paper durch ein *double blind*[6] Verfahren und Verfassung eines Gutachtens
4. Evaluation des Gutachtens durch die Autorinnen und Autoren und Klärung von offenen Fragen und Missverständnissen
5. Reevaluation der Arbeiten und des Gutachtens abhängig von dem Feedback der Autorinnen und Autoren in der vorherigen Phase
6. Diskussion und Entscheidung, ob eine Arbeit akzeptiert und ob dessen Autorinnen und Autoren zur ICSE eingeladen werden.

Bei dem Begutachten der Arbeiten in der 3. Phase wird ein Evaluationsbogen genutzt, der sich historisch über die Jahre entwickelt hat. Dabei sind die Kriterien vom *Program Chair* vorgegeben: [7]

1. *Soundness*, inwieweit werden die Ergebnisse einer Arbeit durch die Anwendung geeigneter empirische Methoden unterstützt
2. *Significance*, wie wichtig sind diese Ergebnisse in dem Feld Software Engineering
3. *Novelty*, wie originell sind diese Ergebnisse.
4. *Verifiability*, inwieweit können die Ergebnisse reproduziert werden, um diese zu verifizieren
5. *Presentation*, inwieweit werden die Standards der ICSE hinsichtlich der Präsentation der Ergebnisse erfüllt

Anhand dieser Bewertungskriterien können sich Autorinnen und Autoren orientieren und ihre Arbeiten entsprechend aufbereiten, indem sie auf diese eingehen und ausreichend beschreiben.

[6] https://conf.researchr.org/track/icse-2021/icse-2021-submitting-to-icse2021–q-a
[7] https://2020.icse-conferences.org/track/icse-2020-papers?Call-for-Papers

Vermutung: Aufgrund des zweiten Bewertungskriterium wird vermutet, dass die Autorinnen und Autoren die Relevanz, sowie die Verallgemeinerbarkeit ihrer Ergebnisse verdeutlichen, und somit auch den größeren Kontext ihrer Arbeit beschreiben.

Die Gutachterinnen und Gutachter bewerten eine Arbeit mit einer ordinalen Skala:
A – I will champion and strongly recommend acceptance
B – I am supportive of acceptance
C – I am weakly not supportive of acceptance but can be convinced otherwise
D – I am not supportive of acceptance. I believe the paper should be rejected

Diese Bewertung kann in den nachfolgenden Phasen angepasst werden, jedoch ist die Bewertung am Ende nicht ausschlagsgebend, sondern der Tatbestand, ob eine Arbeit bei der Konferenz angenommen oder abgelehnt wird. Diese Entscheidung wird in der letzten Phase getroffen. Die Gutachterinnen und Gutachter einer Arbeit diskutieren miteinander, wobei eine Person die Diskussion leitet. Diese Person muss nicht unbedingt selbst die Arbeit begutachtet haben, sondern kann auch ohne Teil des Gutachtensprozesses gewesen zu sein, die Diskussion moderieren. Für den Fall, dass kein Konsens erreicht wird und keine Entscheidung getroffen werden kann, werden weitere Personen zur Diskussion eingeladen und in den Entscheidungsprozess involviert. [8]

Bei der ICSE 2020 bildeten 37 Gutachterinnen und Gutachter das Program Board, die 617 eingereichten Arbeiten evaluiert und bewertet haben. Dabei wurden 129 Arbeiten angenommen und 488 abgelehnt.

2.3 Verwandte Arbeiten

Im Folgendem wird erklärt, wie sich diese Masterarbeit in den Kontext verwandter Arbeiten eingliedert.

Der Grundstein für die Enstehung dieser Masterarbeit ist die Arbeit von **Lutz Prechelt** *"Four presumed gaps in the software engineering research community's knowledge"* [45]
Mittels semi-strukturierten Interviews, Literaturrecherche und einem Beweis-orientierten Ansatz, wurden Wissenslücken in der Software Engineering Forschung aufgedeckt und Ziele für weitere Forschungen gesetzt. Diese Masterarbeit soll dem Ziel eine Taxonomie zu definieren, um eine einheitliche weitreichende Terminologie zu schaffen, näher kommen.

[8] Diese Informationen sind aus den geführten Interviews, welche in Sektion 5.2.2 erörtert werden, hervorgegangen

Andrew Forward und **Timothy C. Lethbridge** haben 2008 in ihrer Arbeit *"A Taxonomy of Software Types to Facilitate Search and Evidence-Based Software Engineering"* eine Taxonomie zu verschiedenen Typen von Software erstellt um empirische Beweise, die auf Software basieren, zuordnen zu können [13]. Dazu haben sie bereits bestehende Taxonomien zu Software Typen analysiert und durch Brainstorming eine eigene Taxonomie erstellt.
Forward und Lethbridge meinen, dass der Softwaretyp einen Kontext darstellt. Doch in dieser Masterarbeit bezieht sich ein Kontext einer empirischen Arbeit auf mehr als den Typ der Software, die untersucht oder vorgestellt wurde, sodass die von dieser Masterarbeit resultierende Taxonomie umfassender ist.

In der Arbeit *"Context in Industrial Software Engineering Research"* 2009 verfasst von **Kai Petersen** und **Claes Wohlin**, wurde eine Checkliste erstellt, um Wissenschaftler und Wissenschaftlerinnen zu unterstützen den Kontext ihrer Arbeiten zu beschreiben [44]. Des Weiteren haben Petersen und Wohlin Literatur der Fachzeitschrift *Empirical Software Engineering* (EMSE) gesichtet, um zu evaluieren inwieweit Autoren und Autorinnen ihre Checkliste erfüllen.
Die Checkliste wurde unterteilt in sechs Kategorien *product*, *processes*, *practices, tools and techniques*, *people*, *organization*, und *market*, welche eine erste mögliche Unterteilung einer Taxonomie darstellen könnten, wie sie auch in dieser Arbeit entwickelt werden soll. Wie die Arbeit von Forward und Lethbridge wird die Arbeit bei der Evaluation der Taxonomie berücksichtigt werden.

Tore Dybå, **Dag I.K. Sjøberg** und **Daniela S. Cruzes** verfassten 2012 ebenfalls eine Arbeit zu dem Thema Kontexte in Software Engineering: *"What works for whom, where, when, and why?: on the role of context in empirical software engineering"* [12]. Sie behaupten, dass in der empirischen Forschung zu Software Engineering, eher nach universalen Beziehungen gesucht werden, die unabhängig von dem Kontext sind. Jedoch machen sie deutlich, dass sich einige Forschungsfragen in unterschiedlichen Kontexten unterschiedlich beantworten lassen. Sie erklären, wie eine Forschungsfrage kontextuell gefragt werden kann und dass die Beschreibung von Kontexten wichtig ist, um Wissen über mehrere Studien hinweg aufzubauen.
Die Verfasser und Verfasserinnen beziehen sich auf die Definition des Begriffes Kontext von **Gary Johns**, der in seiner Arbeit *"The essential impact of context on organizational behavior"* untersucht welchen Einfluss ein Kontext hat und in welchen Dimensionen ein Kontext beschrieben werden kann [23]. Johns forscht dabei jedoch in dem Fachgebiet zum organisatorischem Verhalten.

3 Vorgehen

Zur Anfertigung einer Taxonomie von Software Engineering Kontexten soll analysiert werden, in welchen Kontexten sich eine Arbeit bewegt. Um diese zu identifizieren, wurden innerhalb einer Literaturstudie technische Arbeiten gesichtet. Das genauere Vorgehen wird im Folgendem erläutert, und Forschungsfragen werden aufgestellt.

3.1 Sichtung der Arbeiten

Innerhalb dieser Literaturstudie wurden Arbeiten mehrmals gesichtet. Es wurde ein iterativer Prozess verfolgt, wobei in den Iterationen unterschiedliche Sektionen der Arbeiten untersucht wurden. Dabei wurde eine Arbeit nicht komplett von Anfang bis Ende vollständig gelesen. Um effizienter vorzugehen und Kontexte schneller zu identifizieren, liegt der Fokus auf Sektionen, von denen angenommen wird, dass diese viele Kontexte enthalten.

In dieser Literaturstudie wurden Arbeiten der ICSE gesichtet. Dabei wurde sich nicht nur auf die Konferenz eingeschränkt, sondern auch auf das Veranstaltungsjahr der Konferenz und auf die Art der Arbeiten. Es wurden die technische Arbeiten aus dem Jahr 2020 der ICSE, untersucht, um Kontexte aus der aktuellsten Software Engineering Literatur zu ermitteln. Die Wahl fiel dabei auf die ICSE Konferenz, da die Artikel, die in Rahmen dieser Konferenz veröffentlicht und vorgestellt werden, eine hohe Qualität aufweisen und somit erwartet wird, dass die Autorinnen und Autoren den Kontext ihrer Arbeiten beschreiben.

Außerdem wurde innerhalb dieser Literaturstudie zwischen Arbeiten, die Auszeichnungen oder Abzeichen erhalten haben, und Arbeiten ohne Auszeichnung und Abzeichen unterschieden. Aufgrund folgender Annahmen, wurde auf die Vollständigkeit verzichtet und nur die technischen Arbeiten mit einer Auszeichnung oder einem Abzeichen wurden in dieser Literaturstudie betrachtet.

Annahme: Arbeiten mit Auszeichnungen weisen eine höhere Qualität auf, sodass in diesen Arbeiten mehr über Kontexte und die Verallgemeinerbarkeit der Ergebnisse diskutiert wird.

Annahme: Für Autorinnen und Autoren, die Artefakte ihrer Arbeiten zugänglich machen, ist die Reproduzierbarkeit ihrer Ergebnisse wichtig. Es wird angenommen, dass sie beweisorientierter arbeiten.

Von den 129 technischen Arbeiten der ICSE 2020 wurden an insgesamt 59 eine Auszeichnung und/oder ein Abzeichen verliehen. Zehn Arbeiten haben die Auszeichnung *"ACM SIGSOFT Distinguished Artifact Paper"* und drei Arbeiten die Auszeichnung *"ACM SIGSOFT Distinguished Artifact Award"* erhalten. Außerdem wurden 34 Arbeiten mit dem Abzeichen *"ACM Artifacts Evaluated Reusable"* und 47 Arbeiten mit dem Abzeichen *"ACM Artifacts Available"* ausgezeichnet.
Dabei besteht eine Überschneidung von den ausgezeichneten Arbeiten, wie in der Grafik 3.1 dargestellt.

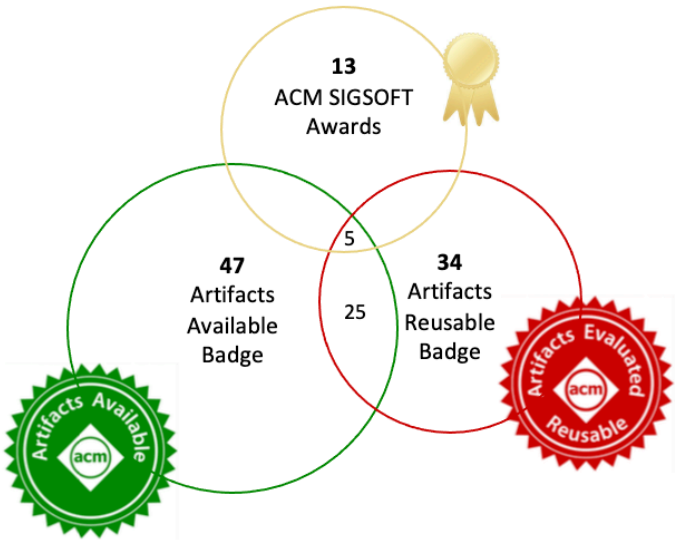


Figure 3.1: 59 Arbeiten mit Auszeichnungen und/oder Abzeichen

In der ersten Iteration basiert die Reihenfolge, anhand welcher die Arbeiten gelesen wurden, auf den erhaltenen Auszeichnungen und Abzeichen. Zuerst wurden die 13 Arbeiten mit der Auszeichnung *"ACM SIGSOFT Distinguished Artifact Paper"* oder mit der Auszeichnung *"ACM SIGSOFT Distinguished Artifact Artifact"* gelesen, danach die 25 Arbeiten mit zwei Abzeichen, die 17 Arbeiten mit dem Abzeichen *"ACM Artifacts Available"* und im Anschluss die 4 Arbeiten mit dem Abzeichen *"ACM Artifacts Reusable"*. Dabei wurden die Arbeiten innerhalb dieser Gruppierungen abhängig von den Präsentationszeitpunkt auf der ICSE gesichtet. Das heißt, dass Arbeiten, dessen Präsentation am 1.Tag der Konferenz gehalten wurden, zuerst gelesen wurden und Arbeiten, die am letzten Tag der Konferenz präsentiert wurden, wurden zuletzt innerhalb der zuvor erläuterten Gruppierung gelesen. Auf der Konferenz wurden die Arbeiten bereits in Themenblöcke gruppiert, sodass ähnliche Themen hintereinander vorgestellt wurden. Dieses zweite Kriterium zur Lesereihenfolge, hat den Vorteil, dass die Arbeiten mit einem ähnlichem Thema, hintereinander gelesen wurden, wenn diese dieselben Auszeichnungen bzw. Abzeichen erhalten haben.
Während dem erstmaligen Sichten einer Arbeit, wurde diese charakterisiert und in Themengebiete eingeordnet, sodass aufgrund dessen in den folgenden Iterationen eine andere Lesereihenfolge bestimmt werden konnte.

3.1.1 Identifizierung von Kontexten

Die Identifizierung der Kontexte wurde durch eine systematische Suche vollzogen. Wie bereits erwähnt, wurde eine Arbeit nicht vollständig von Anfang bis zum Ende gelesen, da der Arbeitsaufwand, der mit dem vollständigem Erfassen der Arbeiten verbunden ist, nicht dem Ziel dieser Masterarbeit entspricht die Kontexte der Arbeiten zu extrahieren und in eine Taxonomie zu überführen.

Annahme: Es wird angenommen, dass die Beschreibungen von Kontexten unterschiedlich häufig an unterschiedlichen Stellen einer Arbeit vorkommen.
Im Folgenden werden weitere Annahmen getroffen, an welchen Stellen welche Kontexte häufiger auftreten. Auf diese Stellen wird sich beim Sichten der Arbeiten und Extrahieren der Kontexte konzentriert.

Annahme: Es wird angenommen, dass die Kontexte in Arbeiten aufgrund verschiedener Zwecke beschrieben werden und das abhängig von diesen Zwecken Kontexte an unterschiedlichen Stellen in einer konkreten Arbeit vorkommen.
Dieser Zusammenhang zwischen dem Zweck eines Kontextes und der Stelle in der Arbeit wird im Weiteren erläutert und in der Tabelle 3.1 aufgeführt.

In dem Titel und in der Einleitung einer Arbeit ist zu erwarten, dass Kontexte zur Charakterisierung des Themenbereich oder zur Definition des untersuchten Problems beschrieben werden.
Des Weiteren können Kontexte bei der Beschreibung einer Technik oder des Vorgehens einer empirischen Studie aufgeführt werden, um Einzelheiten zu Beleuchten und zu Beschreiben in welchem Kontext Daten erhoben wurden.
Bei der Diskussion verwandter Arbeiten, können Kontexte den Zweck haben, die konkrete Arbeit einzugrenzen und von anderen Arbeiten abzugrenzen. An dieser Stelle können Kontexte genannt werden, die das Thema der Arbeit einschränken und somit spezialisieren. Es kann beschrieben werden, in welchen Kontexte die Studie nicht durchgeführt wurde und außerdem kann die Studie oder dessen Ergebnisse in Bezug zu Ergebnissen anderer Arbeiten positioniert werden.
Bei den Schlussfolgerungen, Nennung von Forschungsbeiträgen und bei der Diskussion, wo unterem anderem diskutiert wird, welche Faktoren die Gültigkeit der Arbeit gefährden könnten und wie verallgemeinerbar die Ergebnisse sind, ist zu Erwarten, dass an diesen Stellen viele Kontexte zu finden sind, die sich konkret auf die Verallgemeinerbarkeit und Übertragbarkeit der Arbeit beziehen.

Diese Zuordnung von Vorkommen zu Zweck, wie in der Tabelle 3.1 dargestellt, wird nicht als starre Regel angesehen. Bei der Identifikation von Kontexten, werden unabhängig von den Sektionen, wo diese vorkommen, evaluiert zu welchen Zwecken

die Kontexte gehören. Der selbe Kontext kann mehrmals in einer Arbeit von den Autorinnen und Autoren erwähnt werden.

Zweck	Vorkommen
Thema	Titel, Einleitung, Recherchefragen
Details	Beschreibung, Vorgehen, Datenerhebung
Ein-/Abgrenzung	Diskussion verwandter Arbeiten
Verallgemeinerbarkeit	Schlussfolgerungen, Forschungsbeiträge, Diskussion

Table 3.1: Vorkommen von Kontextfaktoren nach Zweck

Es soll evaluiert werden, ob die Zwecke: Thema, Details, Ein-/Abgrenzung und Verallgemeinerbarkeit ausreichen oder ob weitere Zwecke erkennbar sind, zu welche Autorinnen und Autoren einen Kontext beschreiben. Ebenfalls sollen die folgenden Forschungsfragen beantwortet werden.

Forschungsfragen:

- In wie weit beschreiben Autorinnen und Autoren den Kontext ihrer Arbeit?
- Sind andere Zwecke, als die in Tabelle 3.1 genannten, erkennbar?

Bei dem erstmaligen Sichten der Arbeiten der ISCE 2020, wurde sich bei der Suche nach Kontexten auf drei Zwecke für Kontexte beschränkt: Thema, Ein-/Abgrenzung und Verallgemeinerbarkeit. Demzufolge wurde sich auf die entsprechenden Sektionen in der Arbeit beschränkt, von denen vermutet wird, wo die Kontexte zu diesem Zwecken am Häufigsten vorkommen.
Da es keinen Standard gibt, wie Autorinnen und Autoren die Titel der Sektionen ihrer Arbeiten benennen, wurde sich vor Beginn der Sichtung ein Überblick verschafft, wie die Sektionen einer konkreten Arbeit benannt wurden. Anschließend wurden die Sektionen gelesen, von denen eingeschätzt wurde, dass sich dort Kontexte zu dem gesuchten Zwecken befinden könnten.

Zunächst wurden die Sektionen, die sich auf das Thema der Arbeit beziehen gelesen, um nicht nur Kontexte bezüglich des Themas zu finden, sondern auch ein schnelles Verständnis in das Thema der Arbeit zu erlangen. Anschließend wurden die Sektionen bezüglich der Ein- und Abgrenzung gesichtet, wie die Diskussion verwandter Arbeiten, welche das Thema eingrenzen. Danach wurden die Sektionen bezüglich der Verallgemeinerbarkeit, wie die Schlussfolgerungen und Diskussion gelesen. Dabei wurde ebenfalls analysiert, welchen Beitrag die Arbeit leistet.

Anschließend wurde in der zweiten Iteration mehr auf das Vorgehen, die Datenerhebung und das Design der Studie eingegangen und entsprechende Sektionen in den Arbeiten gelesen, um Kontexte zu dem Zweck Details und zu den neuen Zwecken, die sich aus der Evaluiering der ersten Iteration ergeben haben, zu finden. Daher wurden

in der zweiten Iteration eventuell auch noch Kontexte bezüglich anderer Zwecke als die in der Tabelle 3.1 genannten, extrahiert. Des Weiteren wurden die bereits in der ersten Iteration gefundenen Kontexte evaluiert und geprüft, ob diese doch einem anderen Zweck zugeordnet werden könnten. Dazu wurden die entsprechenden Stellen, in denen ein Kontext innerhalb der ersten Iteration gefunden wurde, erneut evaluiert.

In der dritten Iteration wurden die gefundenen Kontexte evaluiert, indem dessen Vorkommen in einer Arbeit gesichtet wurde und die korrekte Zuordnung zum Zweck erneut überprüft wurde. Außerdem wurden die Kontexte einer Arbeit dahingegend analysiert, ob diese eine Gruppe oder eine Hierarchie bilden. Die Kontexte wurden jedoch nicht nur innerhalb einer Arbeit auf Hierarchien untersucht, sondern auch Arbeitsübergreifend gruppiert. Dies stellt der ersten Versuch zur *Anfertigung einer Taxonomie* dar.

3.1.2 Anfertigung einer Taxonomie

Um eine erste Taxonomie von Software-Engineering Kontexte anzufertigen, müssen die Kontexte, nicht hinsichtlich ihres Vorkommen in der Arbeit oder hinsichtlich ihres Zweckes klassifiziert werden, sondern hinsichtlich ihrer Ähnlichkeit evaluiert und klassifiziert werden. Ähnliche Kontexte, die man unter einen Oberbegriff klassifizieren kann, werden gruppiert. Dabei ist die Gruppierung unabhängig davon, ob die Kontexte sich nur in einer Arbeit befinden oder Arbeitsübergreifend gefunden wurden.

Da während des Sichtungsprozesses der ersten Arbeiten noch keine Oberbegriffe und Gruppierungen bestehen, weil diese fortschreitend ausgearbeitet wurden, mussten diese und dessen zugeordneten Kontexte regelmäßig evaluiert und angepasst werden. Alle bereits zugeordneten Kontexte müssen sich mit dem neu klassifizierten Kontext einer anderen Arbeit ähneln und der Oberbegriff muss zu allen zugeordneten Kontexten passen.
Es kann vorkommen, dass ein Oberbegriff mit einen generalisierten Begriff ersetzt wird, um alle Kontexte zu umfassen, oder dass eine Gruppe auf zwei spezialisierte Gruppen aufgeteilt wird, sodass eine Hierarchie von Kontexten entstehen kann.

3.1.3 Analyse zur Verallgemeinerbarkeit

Neben den bereits in der Sektion 3.1.1 aufgeführten Forschungsfragen zu den identifizierten Kontexten, sollen weitere Forschungsfragen zur Beschreibung der Verallgemeinerbarkeit beantwortet werden.

Forschungsfragen:

- In wie weit diskutieren Autorinnen und Autoren über die Verallgemeinerbarkeit ihrer Ergebnisse?
- Bestehen Lücken in der Diskussion zur Verallgemeinerbarkeit?
- Wie unterscheiden sich die Kontexte zum Zweck der Verallgemeinerbarkeit zu den Kontexten anderer Zwecke?

Zur Beantwortung dieser Forschungsfragen werden die Diskussionen zur Verallgemeinerbarkeit untersucht und die Aussagen der Autorinnen und Autoren genauer betrachtet. Diese Aktivität wird beim Identifizieren der Kontexte bezüglich des Zweckes Verallgemeinerbarkeit durchgeführt.

3.2 Dokumentation

Um eine Übersicht der Ergebnisse innerhalb diesen iterativen Prozesses zur Sichtung der Arbeiten zu ermöglichen, werden die Werkzeuge *MaxQDA* und *Microsoft Excel* zur Dokumentation genutzt.

Mit Hilfe von *MaxQDA* werden die Arbeiten gesichtet und interessante Stellen werden markiert. Dabei werden verschiedene Farben für verschiedene Zwecke genutzt: rot für die Kontexte, gelb für Notizen zum Verständnis, blau für Zitate bezüglich der Verallgemeinerbarkeit, lila für Zitate bezüglich Future Work oder Ergebnisse und grün für die Charakterisierung der Arbeit.

Während der Sichtung einer Arbeit werden Ergebnisse mittels *Microsoft Excel* dokumentiert. Daten zur Charakterisierung einer Arbeit werden festgehalten, Kontexte werden erfasst und Einsichten zu der Analyse der Verallgemeinerbarkeit werden beschrieben.

Um die Analyse der Daten und das Erkennen von Verbindungen zwischen den verschieden Ergebnissen, die in verschiedenen Excel Tabellen dokumentiert sind, zu erleichtern, wurde ein Prototyp zur Datenanalyse und zur Datenvisualisierung entwickelt.
Dieser Prototyp hat den Namen TSEK-Analyzer [9], welches ein Akronym von Taxonomie von Software Engineering Kontexten ist. Es wurde durch das Framework Angular umgesetzt und durch *GitHub Pages* bereitgestellt. Die genaue Umsetzung des Prototyps wird in dem nächsten Kapitel in der Sektion 4.5 Implementierung des Prototyps erläutert.

[9] https://stephaniehohenberg.github.io/TSEK-Analyzer/

4 Durchführung

Im Folgenden wird die Realisierung des im vorherigen Kapitel beschriebene Vorgehen dokumentiert. Die Ergebnisse und Schlussfolgerungen sollen die Beantwortung der Forschungsfragen unterstützen.

4.1 Charakteristiken der Arbeiten

Die Arbeiten wurden zum Einen auf die Art der durchgeführten Studie hin charakterisiert und zum Anderen zum Vorgehen und den Methoden, die zur Evaluierung der Ergebnisse angewandt wurden. Dabei wurde lediglich die explizite Erwähnung von den Charakteristiken einbezogen und nicht nach impliziten Andeutungen gesucht.

Diese Charakterisierung wurde überwiegend in der ersten Iteration der Sichtung der Arbeiten durchgeführt, in welcher unter Anderem die Stellen zu dem Zweck Thema (Titel, *Abstract*, Einleitung und Forschungsfragen) sowie das Fazit der Arbeiten gelesen wurde. In der zweiten Iteration, in welcher Sektionen bezüglich des Vorgehens, der Datenerhebung und des Design der Studie gelesen wurden, wurde die Charakterisierung ergänzt, falls es sein könnte, dass Autorinnen und Autoren, sich mehr über die angewandten Methoden und ihr Vorgehen in den Details einer Arbeit äußern, als in der Einleitung oder im Fazit der Arbeit. Durch die zweimalige Ausführung dieses Vorgehens und das gründliche Sichten verschiedener Stellen der Arbeiten wird erwartet, dass die Charakteristiken gut erkannt wurden.

In Tabelle 4.1 und Tabelle 4.2 wurden die Ergebnisse zu den Charakteristiken aufgelistet.

#	Art der Studie
13	Empirische Studie
6	Umfassende Studie
4	Pilotstudie
3	Explorative Studie
3	Nutzerstudie
2	Fallstudie
1	Feldstudie

Table 4.1: Charakterisierung zur Art der Studie der 59 technischen Arbeiten

Insgesamt haben 23 Arbeiten angegeben eine Studie durch zu führen. Die häufigste Art ist die empirische Studie mit 13 Arbeiten und die zweit Häufigste die umfassende Studie mit 6 Arbeiten. Dabei besteht eine Überschneidung von 2 Arbeiten, die angegeben haben, sowohl eine empirische, sowie eine umfassende Studie durchzuführen. Die Autorinnen und Autoren von insgesamt 8 Arbeiten haben die Art ihrer Studie mehrmals definiert.
Es fällt auf, dass mehr als die Hälfte der Arbeiten, insgesamt 36 Arbeiten, nicht angegeben haben, welche Art von Studie durchgeführt wurde.

#	Vorgehen der Arbeit
42	Präsentation eines Ansatzes
24	Experiment
	> davon 3 kontrollierte Experimente
	> davon 1 psychologische Experimente
	> davon 1 online Experimente
22	Untersuchung des *States-of-Arts* bzw. Vergleich mit einen anderen Arbeit
20	Benchmark
21	Inspektion
7	Interview
	> davon 2 strukturierte Interviews
	> davon 1 semi-strukturierte Interviews
6	Umfrage
1	Contextual Inquiry
1	Simulation

Table 4.2: Charakterisierung des Vorgehens der 59 technischen Arbeiten

Bei 42 der 59 gesichteten Arbeiten wurde ein Ansatz, eine Technik oder eine Methode präsentiert. In 24 Arbeiten wurde ein Experiment durchgeführt und in 22 Arbeiten wurde die States-of-Arts untersucht oder ein Vergleich wurde vollzogen. In 20 Arbeiten ist eine Benchmark angewandt worden und in 21 Arbeiten wurde eine manuelle Analyse bzw. eine Inspektion durchgeführt worden. In 7 Arbeiten wurden Interviews geführt, in 6 Arbeiten eine Umfrage und lediglich in einer Arbeit wurde eine Contextual Inquiry und in einer anderen eine Simulation durchgeführt.

Schlussfolgerung: Durch die überwiegende Mehrheit von technischen Arbeiten, die einen Ansatz, eine Technik oder eine Methode präsentiert, kann geschlussfolgert werden, dass die Ergebnisse dieser Masterarbeit sich mehr im Kontext solcher Arbeiten bewegen, und weniger im Kontext von technischen Arbeiten, die bereits bestehende Forschungen evaluieren.
Inwieweit das einen Einfluss auf die Relevanz oder die Verallgemeinerbarkeit dieser

Masterarbeit hat, wird in Kapitel 6 Sektion 6.3 der Diskussion zur Relevanz und Verallgemeinerbarkeit geklärt.

Da nicht alle Arbeiten der ICSE 2020 gesichtet wurden, kann nicht geschlussfolgert werden, ob auf der ICSE 2020 ein Trend besteht, dass eher Arbeiten, die einen Ansatz, eine Technik oder eine Methode vorstellen, angenommen werden. Da der Bereich der Software Engineering jedoch ein sehr angewandter Bereich der Informatik ist, ist dies durchaus anzunehmen.

4.2 Analyse zur Verallgemeinerbarkeit

Im Folgenden wird untersucht, inwieweit Autorinnen und Autoren über die Verallgemeinerbarkeit ihrer Ergebnisse diskutieren. Diese kritische Analyse bezüglich der Verallgemeinerbarkeit wurde in der ersten Iteration des Sichtungsprozesses durchgeführt. Dabei wird zunächst nur kurz auf die gefundenen Kontexte zur Verallgemeinerbarkeit eingegangen, da in der nächsten Sektion 4.3 dieses Kapitels die identifizierten Kontexte näher erläutert werden. Mittels der IDs der Arbeiten, die sich nach dem regulären Ausdruck [ab][0-9]{3,3} bilden, wird dabei auf die untersuchten Arbeiten referiert.

Bei der kritischen Analyse bezüglich der Verallgemeinerbarkeit wurde zunächst die Sektion *Threats to Validity* betrachtet. Der Fokus lag dabei auf der Diskussion und Aussagen über die externe Gültigkeit einer Arbeit.
Von den 59 Arbeiten haben 42 Arbeiten eine Sektion *Threats to Validity*. In 19 Arbeiten (32%) wurde die Sektion *Threats to Validity* in *internal validity*, *external validity* und *construct validity* gegliedert. In 9 Arbeiten (15%) wurde zwischen *internal validity* und *external validity* unterschieden. In wiederum 9 Arbeiten (15%) wurde nur die *internal validity* betrachtet, in einer Arbeit wurde lediglich die *external validity* und in einer anderen lediglich die *construct validity* beschrieben. Drei Arbeiten beschreiben lediglich die Einschränkungen einer Arbeit anstatt die Gültigkeit der Arbeit.

In 17 Arbeiten (29%) fehlt die Sektion *Threats to Validity*, wobei in drei dieser Arbeiten die *Threats to Validity* als eine Passage innerhalb einer anderen Sektion erläutert wurde, statt in einer eigenen Sektion.

Schlussfolgerung: Diese uneinheitliche Beschreibung der *Threats to Validity* lässt vermuten, dass es keine Standards zur Beschreibung der internen und externen Gültigkeit, sowie der Konstruktvalidität einer Arbeit gibt.

Da Aussagen und Kontexte zur Verallgemeinerbarkeit auch in anderen Sektionen vorkommen können, wurden, wie in Kapitel 3 erwähnt, ebenfalls die folgenden Sektionen untersucht: *Conclusion*, *Limitations*, *Future Work* und *Discussion*. In 16 Arbeiten (27%) befindet sich eine Sektion *Limitations* und in 32 Arbeiten (54%) befindet sich eine Sektion oder eine Passage zur *Future Work*.

Es fällt auf, dass nicht nur mehr über die interne Gültigkeit gesprochen wird (in 37 Arbeiten (63%)), als über die externe Gültigkeit (in 29 Arbeiten (49%)), aber auch dass die Passage zur externen Gültigkeit oft in nur wenigen Zeilen beschrieben.

In zwei Arbeiten lag der Fokus nicht auf der Verallgemeinerbarkeit (a654, a691) und in drei Arbeiten wird ausgesagt, dass man keine generellen Schlüsse ziehen kann oder möchte (b273, b209, b359). Wohingegen in 11 Arbeiten (19%) gar keine Aussagen zur Verallgemeinerbarkeit gefunden wurden (b073, a481, b024, b410, a542, b049, a824, a493, b483, a258, b086).

> *'While desirable, generalizability was not the main focus of this study'* (id: a654) [5]
>
> *'We cannot claim, the results for each case study will generalize for other kinds of programs (or classifiers) written in other languages or from other domains.'* (id: b273) [38]

In acht Arbeiten (14%) werden vage Aussagen zur Verallgemeinerbarkeit gemacht, dass die Ergebnisse sich möglicherweise nicht generalisieren lassen (a457, a210, b447, a335, b098, a271, a678, b435). In zwei von diesen Arbeiten und in vier Weiteren wird die Evaluation der Verallgemeinerbarkeit auf zukünftige Arbeiten verschoben (a678, b435, a949, a765, a125, a714).

> *'Our study is limited to the evaluated Android apps and our results may not generalize beyond the evaluated apps.'* (id: a457) [62]
>
> *'Nevertheless, our results may not generalize to all in-the-wild programming; we leave a more thorough investigation to future research.'* (id: a678) [28]

In 27 Arbeiten (46%) wird die Verallgemeinerbarkeit mit der Repräsentativität der evaluierten Subjekte verknüpft. Dabei wird unter Anderem auf die Beliebtheit von genutzten Benchmarks oder die Diversität von Teilnehmenden hingewiesen (a087, a322, a309, b171, b110, a615, a075, a530, a949, a765, b122, b135, b509, b098, a271, a284, b061, a234, a469, a186, a666, a566, b572, b183, a409, b348, a641).

> 'Our use of this well-known and widely-used benchmark of real-world defects aims to ensure our results generalize.' (id: a087) [24]

> ’We chose the same task used in a former study [42], which simulated a new problem in a real scenario. Regarding participants, we covered different levels of academic experience (by including Bachelors, Masters, and PhD students) but with less professional experience compared to [42].’ (id: a666) [15]

Lediglich in drei Arbeiten wird mit Begründung ausgesagt, dass die Ergebnisse verallgemeinerbar sind (b535, b159, a752). In zwei weiteren Arbeiten wurden zusätzliche Experimente durchgeführt, um somit zu beweisen, dass die Ergebnisse verallgemeinerbar sind (b261, b459). Im Gegensatz dazu werden in drei Arbeiten die Ergebnisse ohne Begründungen oder Erläuterungen als verallgemeinerbar dargestellt (a026, b248, a986).

> *’Although our study is conducted exclusively in Microsoft, we believe that our failure categories are prevalent and most of our results can be generalized to other DL platforms. The main reason is that most deep learning programs executed on Philly use the common programming paradigm (e.g., Python language, stochastic algorithms, frameworks, and toolkit libraries) and target at common applications (e.g., image/video/speech recognition and NLP). Besides, the hardware, system architecture, and job submission/execution mechanism of Philly are widely adopted [9, 23, 47].’* (id: b159) [68]

> ’The evaluations are conducted on sentiment analysis task. The proposed TRADER, however, is not application specific.’ (id: a986) [53]

In der Tabelle 4.3 werden die erwähnten Zahlen zusammengefasst, wobei jedoch nicht die Überschneidungen erkennbar sind.

Schlussfolgerung: An den untersuchten Stellen *Threats to Validity*, *Conclusion*, *Limitations*, *Future Work* und *Discussion* werden nur wenige Aussagen zur Verallgemeinerbarkeit getätigt und innerhalb der Hälfte der Arbeiten verlassen sich die Autorinnen und Autoren auf die Repräsentativität der ausgewählten Subjekte, anstatt über die Verallgemeinerbarkeit zu diskutieren und zu argumentieren.
Die gefundenen Kontexte mit dem Zweck Verallgemeinerbarkeit unterscheiden sich zu den identifizierten Kontexten anderer Zwecke in der Hinsicht, dass diese in vagen Aussagen gefunden wurden und diese Aussagen nicht immer begründet oder erläutert wurden.

Über die Gründe, warum Autorinnen und Autoren nicht ausreichend über die Verallgemeinerbarkeit ihrer Ergebnisse diskutieren kann nur gemutmaßt werden.

#	
42	Sektion *Threats to Validity* vorhanden
	davon 19: Aufteilung in internal, external und construct validity
	davon 9: Aufteilung in internal und external validity
	davon 9: Beschreibung der internen Validität
	davon 1: Beschreibung wie der externen Validität
	davon 1: Beschreibung wie der construct Validität
	davon 3: Beschreibung der Einschränkungen der Ergebnisse
17	Keine Sektion *Threats to Validity* vorhanden
	davon 3: Beschreibung der *Threats to Validity* als Passage in einer anderen Sektion
16	Sektion *Limitations* vorhanden
32	Sektion oder Passage zur *Future Work* vorhanden
2	Kein Fokus auf Verallgemeinerbarkeit
3	Keine Schlussfolgerungen zur Verallgemeinerbarkeit möglich
11	Keine Aussagen zur Verallgemeinerbarkeit
8	Vage Aussagen zur Verallgemeinerbarkeit
6	Weitere Forschung nötig zur Evaluation der Verallgemeinerbarkeit
27	Verallgemeinerbarkeit durch Repräsentativität der Subjekte
3	Aussagen zur Verallgemeinerbarkeit mit Erläuterungen
3	Aussagen zur Verallgemeinerbarkeit ohne Erläuterungen
2	Durchführung weiterer Experimente zur Evaluation der Verallgemeinerbarkeit

Table 4.3: Zusammenfassung der Analyse der Diskussion zur Verallgemeinerbarkeit

Vermutungen: Innerhalb 28 Arbeiten (47%) wird ausgesagt, dass eine erstmalige Studie durchgeführt wurde oder eine neue Methode, Technik oder Ansatz entwickelt wurde und wie bereits in der vorherigen Sektion erkannt, präsentieren 42 der 59 Arbeiten einen Ansatz, eine Methode oder eine Technik. Daher ist es möglich, dass die Autoren und Autorinnen, es nicht als relevant ansehen die Verallgemeinerbarkeit zu diskutieren. Zum Anderen könnte es den Autoren und Autorinnen selbstverständlich erscheinen, inwieweit ihre Ergebnisse verallgemeinerbar sind und es dem Lesenden offensichtlich sein sollte.
Auch kann es sein, dass die Evaluation der Verallgemeinerbarkeit weitere Experimente benötigt, die zu zeitaufwändig für die Arbeit waren, oder dass der Fokus der Studie eben nicht auf die Verallgemeinerbarkeit lag, wie auch in zwei Arbeiten ausgesagt wurde.

Dies wirft die Frage auf, inwiefern eine Arbeit als relevant eingeschätzt wird, wenn die Verallgemeinerbarkeit und somit der größere Kontext, auf den die Ergebnisse einer Arbeit übertragbar sind, unzureichend beschrieben werden.

Vermutung: Es kann sein, dass die Einschätzung zur Relevanz mehr von der Beschreibung der Motivation einer Arbeit abhängt, als von der Beweisführung zur Verallgemeinerbarkeit.

4.3 Analyse der Kontexte

Neben den bereits in Kapitel 3 Sektion 3.1.1 erläuterten Zwecken *Thema*, *Details*, *Ein-/Abgrenzung* und *Verallgemeinerbarkeit*, wurden zwei weitere Zwecke *Erweiterung* und *Anwendung* erkannt und den identifizierten Kontexten zugeordnet.

In der Analyse zur Verallgemeinerbarkeit ist aufgefallen, dass in ein paar Arbeiten die Evaluation der Verallgemeinerbarkeit auf zukünftige Arbeiten verschoben wurde. Dabei wurden Kontexte identifiziert, die jedoch nicht innerhalb der Arbeit gelten, sondern erst durch weitere Forschung validiert werden. Daher wurde der Zweck *Erweiterung* als weitere Zweck untersucht. Dieser wurde in neun Arbeiten entdeckt.

> 'For simplicity, Holmes works on **single-argument tests with String or primitive arguments**. While this is sufficient for the defects in Defects4J benchmark, this limitation will need to be lifted for **tests with multiple arguments**.' (id: a087) [24] - Zweck Eingrenzung und Erweiterung

Außerdem wurde der Zweck *Anwendung* eingeführt für Kontexte, auf dessen die Ergebnisse einer Arbeit anwendbar sind. In vielen Arbeiten wird zu Beginn die Motivation der Arbeit erläutert, dabei werden Anwendungsgebiete und Domänen, sowie Subjekte auf denen die Ergebnisse anwendbar sind, beschrieben. In 42 Arbeiten wurden Kontexte zu diesem Zweck gefunden.
Im Vergleich dazu wurden Kontexte zum Zweck Verallgemeinerbarkeit in nur 19 Arbeiten identifiziert. Dies könnte die Vermutung bestätigen, dass die Relevanz einer Arbeit eher anhand der Beschreibung der Motivation eingeschätzt wird, als von der Beweisführung der Verallgemeinerbarkeit, da die Autorinnen und Autoren diese mehr beschreiben und sich mehr Kontexte identifizieren lassen.
Kontexte zur Anwendung lassen sich jedoch nicht nur bei der Motivation finden, sondern auch bei Sektionen zum Design der ausgeführten Experimenten und zu der Analyse der Ergebnisse.

> 'Smart contracts are programs that provide **security-critical applications** like financial transactions, but they are vulnerable to various security attacks' (id: a542) [21] - Zweck Anwendung

> 'Threats to external validity include the focus on two languages as instances of static and dynamic typing, so results may not generalize beyond **Java** and **Python**.' (id: a210) [35] - Zweck Verallgemeinerbarkeit

> ’We believe that the proposed approach is generally applicable to supporting other testing types such as **security** and **performance testing**, since [...]’ (id: b535) [57] - Zweck Verallgemeinerbarkeit

Es wurden mehr Kontexte zur Abgrenzung als zur Eingrenzung identifiziert. Die Kontexte zur Eingrenzung wurden dabei überwiegend am Anfang einer Arbeit identifiziert und die Kontexte zur Abgrenzung in der Sektion zu den verwandten Arbeiten. Oft waren in diesen Arbeiten bereits hierarchische Strukturen zwischen den Kontexten deutlich.
Unabhängig vom Zweck der Kontexte wurden insgesamt innerhalb 47 Arbeiten (80%) Gruppierungen erkannt, was den nächsten Schritt, die Anfertigung der Taxonomie erleichtert hat.

Insgesamt wurden innerhalb der 59 gesichteten Arbeiten 673 Kontexte identifiziert. In den Tabellen 4.4 und 4.5 ist erkennbar, wie viele Kontexte zu welchem Zweck vorkommen. Es fällt auf, dass die meisten Kontexte zu den Zwecken Details (28%), Anwendung (24%) und Abgrenzung (20%) zugeordnet worden. Die meisten Kontexte kommen dabei in den folgenden Sektionen vor: Einleitung (24%), Abstract (17%) und Related Work (16%).
Die Annahme aus Sektion 3.1.1, dass Kontexte abhängig von ihren Zweck an verschieden Stellen einer Arbeit vorkommen, lässt sich bestätigen. Besonders am Anfang einer Arbeit, im Titel, Abstract und der Einleitung lassen sich viele Kontexte finden. Dabei muss beachtet werden, dass Kontexte, die bereits gefunden wurden, nicht nochmal dokumentiert wurden, falls diese ebenfalls an anderen Stellen zu einen anderen Zweck erneut erwähnt wurden.

Zweck	Thema	Details	Eingrenzung	Abgrenzung
insgesamt	83 (12%)	191 (28%)	54 (8%)	136 (20%)
Titel	**47**	2	**13**	-
Abstract	**26**	**47**	**11**	3
Einleitung	8	**78**	8	**28**
Hintergrund	-	5	-	5
Related Work	-	6	7	**91**
Daten Vorbereitung	-	4	3	-
Analyse der Ergebnisse	-	3	-	-
Experiment Design	-	**36**	7	3
Limitations	-	-	2	5
Threats to Validity	-	6	2	1
Evaluation	-	2	-	-
Future Work	-	-	-	-
Diskussion	1	-	-	-
Schlussfolgerungen	-	-	1	-
Implizit	1	2	-	-

Table 4.4: Identifizierte Kontexte anhand Vorkommen und Zweck

Zweck	Anwendung	Verallgemeinerbarkeit	Erweiterung	ings
insgesamt	163 (24%)	30 (5%)	16 (2%)	
Titel	**17**	-	-	79
Abstract	**24**	-	3	114
Einleitung	**39**	1	-	162
Hintergrund	2	3	-	15
Related Work	1	-	4	109
Daten Vorbereitung	1	-	-	8
Analyse der Ergebnisse	**18**	-	-	21
Experiment Design	**34**	-	-	80
Limitations	2	1	1	11
Threats to Validity	8	**20**	3	40
Evaluation	7	-	-	9
Future Work	-	-	2	2
Diskussion	1	-	2	4
Schlussfolgerungen	1	5	1	8
Implizit	8	-	-	11

Table 4.5: Identifizierte Kontexte anhand Vorkommen und Zweck

4.4 Anfertigung der Taxonomie

Um eine Taxonomie anzufertigen, wurden die Kontexte Arbeitsübergreifend analysiert, gruppiert und für die Gruppen geeignete Oberbegriffe gefunden. Innerhalb einiger Arbeiten konnten Kontexte bereits gruppiert werden. Daher wurde analysiert, ob Kontexte anderer Arbeiten innerhalb dieser Gruppen zugeordnet werden können oder Gruppierungen von Kontexten verschiedener Arbeiten zusammengelegt werden können.

Es haben sich sechs Taxonomien mit folgenden Oberbegriffen gebildet, die auch als Dimensionen bezeichnet werden können: Software Engineering Aktivitäten, Software Werkzeuge, Anwendungen, Daten, Human Aspects, arbeits-spezifische Kontexte.
Dabei sind 74 Oberbegriffe entwickelt worden und 34 Kontexte fassen 98 identische Kontexte zusammen.

Wurzel	Knoten	Zwischenknoten	Blätter	Ebenen
SE Aktivitäten	193	48	144	7
Software Werkzeuge	82	14	67	3
Anwendungen	99	23	75	6
Daten	36	8	27	3
Human Aspects	63	12	50	4
arbeits-spezifische Kontexte	81	10	70	3
insg	554	115	433	

Table 4.6: Zahlen zu den Taxonomien

Die sechs Taxonomien werden in den Grafiken 7.1 bis 7.9 im Anhang dargestellt. Da die Darstellung in einem Din A4 Format unzureichend ist, sind die Taxonomien innerhalb des Werkzeuges TSEK-Analyzer interaktiv visualisiert worden.[10]

Die makroskopischen Beziehungen zwischen den Arbeiten wurden dabei nicht in dem Werkzeug visualisiert, da der Fokus auf die Visualisierung der kompletten Taxonomie lag, sodass Informationen zu den Arbeiten nicht integriert wurden.
Zusätzlich könnte das Werkzeug um Verknüpfungen erweitert werden, sodass ähnliche Kontexte, die in verschiedenen Bäumen dargestellt wurden, verbunden werden und eine explorative Navigation möglich wäre. Ein Beispiel für diesen Fall wäre:

[10] https://stephaniehohenberg.github.io/TSEK-Analyzer/

- *SE Activities* > *Security Assurance* > *Security Attacks* - siehe Grafik 7.1
- *SE Activities* > *Fault Localization / Detection* > *Issues, Defects, Failures* > *Security Vulnerabilities* - siehe Grafik 7.3
- Human Aspects > *Human Values* > Security - siehe Grafik 7.6

Vermutung: Da sich mehrere solcher Beispiele finden lassen, kann vermutet werden, dass eine Taxonomie möglicherweise nicht die beste Darstellung ist um Software Engineering Kontexte lediglich in hierarchischen Beziehungen zu repräsentieren.

Jedoch können durch diese Taxonomie Trends und Lücken erkannt werden.
In der Dimension Software Engineering Aktivitäten fällt auf, dass der Knoten zur Softwareentwicklung weit aus weniger ausgearbeitet ist, als der Knoten zum Testen. Die Knoten zum Testen, Bug Handling und Fault Localization / Detection und ihre Unterknoten sind so groß, dass sie in eigenen Teilbäumen in Grafiken dargestellt wurden. Daher kann geschlussfolgert werden, dass auf der ICSE 2020 ein Trend bezüglich dieser Themen besteht, dass Autorinnen und Autoren mehr in dieser Richtung forschen und mehr Arbeiten bezüglich dieser Themen auf der ICSE 2020 akzeptiert wurden.
In der Dimension Anwendungen lässt sich die Lücke unter den mobilen Anwendungen erkennen. Es wurden Arbeiten gesichtet, die sich lediglich mit Android Anwendungen beschäftigen. Der Kontext dieser Arbeiten wird eingegrenzt und es wurde keine Stellung dazu genommen, ob die Ergebnisse sich auf iOS Anwendungen verallgemeinern lassen oder nicht.

Bei der Taxonomie zu den arbeits-spezifischen Kontexten, welche zum Teil in der Grafik 7.9 abgebildet ist, wurden Oberbegriffe eingeführt, jedoch wurde darauf verzichtet die Kontexte auf ihre Gleichheit zu überprüfen und Zitate wurden beibehalten, da die Kontexte sich meist durch ein kleines Detail unterscheiden oder ein Kontext sehr spezifisch für eine Arbeit ist. Außerdem ist die Skalierung, beispielsweise die Einordnung in kleine und große Projekte, nicht immer klar definiert.

Kontexte von vier Arbeiten konnten nicht in einer der sechs Dimensionen der Taxonomie integriert werden. Diese sind in der Grafik 7.10 im Anhang abgebildet.

4.5 Implementierung des Prototyps: TSEK-Analyzer

Um die Ergebnisse zu den verschieden Aspekten, die in den vorherigen Sektionen vorgestellt wurden, miteinander zu verknüpfen und Zusammenhänge zu erkennen, wurde, wie bereits im vorherigen Kapitel 3 in der Sektion 3.2 zur Dokumentation kurz erwähnt wurde, der Prototyp *TSEK-Analyzer* entwickelt. Dieser ist jedoch nicht nur

zur Dokumentation der Ergebnisse gedacht, sondern soll überwiegend die Datenanalyse und die Visualisierung der angefertigten Taxonomie unterstützen.

Der Prototyp besteht aus verschiedenen Komponenten:

1. Einer Übersicht der einzelnen Arbeiten und dessen Ergebnisse bezüglich der Charakterisierung, der Kontexte, sowie der Verallgemeinerbarkeit
2. Einen Graph zur hierarchischen Visualisierung der Kontexte innerhalb einer interaktiven Taxonomie
3. Einen Filter, um die angezeigten Daten in der Übersicht einzuschränken

Ein Screenshot des Prototyps und dessen Komponenten ist in der Grafik 4.1 zu sehen. Des Weiteren wurde der Prototyp deployed[11] und dessen Code ist auf GitHub[12] verfügbar.

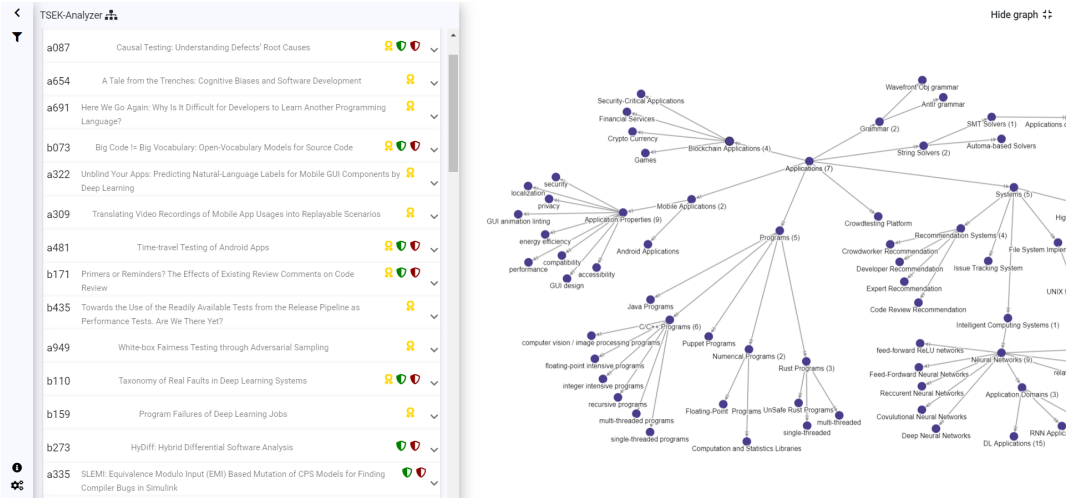


Figure 4.1: Screenshot des Prototyps TSEK-Analyzer mit der Übersichtskomponente (links) und der Graphkomponente (rechts)

Der Prototyp wurde mit Hilfe des Open-Source Frameworks Angular 11 implementiert, welches von dem Unternehmen Google Inc. seit 2009 entwickelt wird und zur Erstellung von Webanwendungen dient.
Eine Angular Webanwendung besteht aus Modulen, welche Komponenten und Services beinhalten. Die Services implementieren die Businesslogik und werden von anderen Services und den Komponenten genutzt. Jede Komponente ist mit einem HTML-Template gekoppelt, wobei die Komponente dem HTML-Template Inhalte dynamisch zur Verfügung stellt und auf Events reagiert. Dabei wird eine Angular Webanwendung in HTML und JavaScript oder eine Sprache, die zu JavaScript transponierbar ist, realisiert, wie in diesem Fall TypeScript, eine von Microsoft entwickelte Programmiersprache, die JavaScript um Typen, Module, Klassen, Interfaces, Generics

[11] https://stephaniehohenberg.github.io/TSEK-Analyzer/
[12] https://github.com/StephanieHohenberg/TSEK-Analyzer/tree/master/project

und viele weitere nützliche Erweiterungen ergänzt.

Des Weiteren wurde die Bibliotheken *Cytoscape*[13] und Angular Material[14] eingebunden.

Die JavaScript-Bibliothek *Cytoscape* dient zur Visualisierung und Analyse von Daten als Graphen und setzt Algorithmen der Graphentheorie um. Diese Bibliothek wurde am *Donnelly Centre der University of Toronto* entwickelt, wo an Zellen und Biomolekülen geforscht wird. Somit wurde Cytoscape ursprünglich genutzt um biologische Daten und zelluläre Mechanismen zu visualisieren. Jedoch kann es in verschiedenen Bereichen eingesetzt werden, wie zur Darstellung von sozialen Netzwerken oder Transportsystemen oder wie im vorliegenden Fall, zur Darstellung der Kontexte innerhalb einer interaktiven Taxonomie.
Cytoscape bietet viele Funktionalitäten, um die Visualisierung von Daten zu erleichtern. Unter anderem stehen verschiedene Optionen zur Verfügung den Graphen zu gestalten und zu animieren. Es werden Funktionen zur Manipulation und zur Interaktion mit dem Graphen unterstützt und es werden verschiedene Möglichkeiten geboten, den erzeugten Graph zu traversieren.

Die Angular-Bibliothek Angular Material stellt vorgefertigte Komponenten mit einem konsistenten Design zur Erstellung einer Angular Webanwendung zur Verfügung. Wie das Angular Framework wurde diese Bibliothek von Google Inc. entwickelt. Daher lehnt es an dem Google Material Design[15] an. Zu den verfügbaren Komponenten zählen nicht nur Formularfelder, wie Eingabefelder, Buttons und Checkboxen, aber auch Komponenten zur Strukturierung und zum Layout der Anwendung und Komponenten, wie Modale oder Expansion Panels, die mit eigenen Inhalten und Daten gefüllt werden können. Dabei sind die Komponenten mit allen modernen Webbrowsern kompatibel.

Die Webanwendung verfügt über vier Services:

- *PaperService* für Zugriffs- und Filter-Funktionen bezüglich der Arbeiten
- *ContextService* für Zugriffs- und Filter-Funktionen bezüglich der Kontexte
- *FilterService* zur Verwaltung der Filtereinstellungen und zur Kommunikation zwischen der Filter-Komponente und den Komponenten zur Übersicht der Arbeiten und zur Graphvisualisierung, sodass jede Komponente, die gefilterte Daten darstellt, stets die aktuellen Filtereinstellungen hat
- *VisibilityService* für das Aus- und Einblenden von Daten, um die Informationsdichte zu reduzieren

[13] https://cytoscape.org/ [14]
[14] https://material.angular.io/
[15] https://material.io/design

Wie bereits erwähnt sind die primären Nutzungsszenarien des Prototypes die Datenanalyse und die Datenvisualisierung. Jedoch kann der Prototyp um weitere Nutzungsszenarien erweitern werden.
Der Prototyp liest lediglich Daten aus und stellt diese dar. Das Nutzungsszenario zur Dokumentation könnte daher um Schreiboperationen, welche über GUI-Komponenten ausgeführt werden können, erweitert werden. Dies hätte den Vorteil, dass die Dokumentation in Microsoft Excel und eine Übertragung der Daten in das Werkzeug nicht von Nöten wäre und somit eine potentielle Fehlerquelle vermieden werden könnte.
Die Daten wurden ebenfalls während der Erstellung der Taxonomie angepasst, um identische Kontexte von verschiedenen Arbeiten zusammenzufassen, paper-übergreifende Kontexte einzuführen und Oberbegriffe in der Taxonomie zu bilden. Dies hätte ebenfalls durch eine Funktionalität im Prototypen erleichtern werden können.
Zusätzlich zu den Schreiboperationen könnte eine Versionierung der Ergebnisse sinnvoll sein, um nachvollziehen zu können, welche Ergebnisse und welche Kontexte in welcher Iteration des Leseprozesses erschlossen wurden. Durch die Möglichkeit eingetragene Ergebnisse durch Tags, wie Iteration 1 bis Iteration 3 oder Evaluierung, zu versionieren, könnte das Vorgehen und der Leseprozess besser evaluiert werden.

Da der Umfang dieser Masterarbeit begrenzt ist, wurden diese potentiellen Erweiterungen nicht umgesetzt. Außerdem wurde der Code des Prototypen nicht getestet und ebenfalls wurde keine Code Review durch eine weitere Person durchgeführt, um den Code auf dessen Qualität und Korrektheit zu prüfen. Jedoch wurden Usability Tests durchgeführt, um den Prototypen zu evaluieren. Die genaue Vorgehensweise und die Ergebnisse dieser Evaluierung werden im nächsten Kapitel in der Sektion 5.3 vorgestellt.

Vermutung: Der Prototyp mit der Visualisierung der Taxonomie kann auch von anderen Personengruppen innerhalb verschiedenen Nutzungsszenarien genutzt werden.
Wie relevant die Taxonomie für verschiedene Gruppen in der Welt des Software Engineerings sein könnte, wird in der Sektion 5.2 evaluiert, sodass an dieser Stelle nicht näher darauf eingegangen wird.

5 Evaluation

Im Folgendem wird das Vorgehen dieser Masterarbeit, die Relevanz, der Prototyp *TSEK-Analyzer* und die angefertigte Taxonomie evaluiert.

5.1 Evaluation des Vorgehen

Im Folgendem wird das Vorgehen, welches in Kapitel 3 beschrieben wurde, evaluiert. Dabei wird besonders auf die Literaturstudie und die Dokumentation eingegangen.

In der vollzogenen Literaturstudie wurde lediglich eine Auswahl der 129 technischen Arbeiten der ICSE gesichtet. Der Fokus lag auf Arbeiten, die eine Auszeichnung und/oder ein Abzeichen erhalten haben, da unter Anderem angenommen wurde, dass diese eine höhere Qualität aufweisen. In Rahmen dieser Arbeit wurde nicht untersucht, ob diese Annahme korrekt ist und, ob Arbeiten ohne Auszeichnung und ohne Abzeichen eine schlechtere Qualität aufweise.
Die beiden Abzeichen *Artifacts Available* und *Artifacts Reusable* beziehen sich auf die Artefakte einer Arbeit. Doch nicht jede Arbeit weist Artefakte auf und für Arbeiten qualitativer Natur, in welchen sensitive Daten gesammelt wurden, kann die Veröffentlichung der Daten aufgrund Datenschutzverordnungen schwieriger sein, als für Arbeiten die beispielsweise GitHub Repositories untersucht haben, die bereits öffentlich zugänglich sind. Somit erfordert der Erhalt eines Abzeichen für Arbeiten mit sensitiven Daten einen höheren Aufwand, sodass diese weniger in dieser Literaturstudie repräsentiert werden. Aus der 4.1 geht hervor, dass technische Arbeiten, die einen neuen Ansatz vorstellen, repräsentativer in dieser Literaturstudie sind. Um Nachteile diesbezüglich entgegen zu wirken, müssten die restlichen technischen Arbeiten der ICSE 2020 gesichtet und in der Taxonomie integriert werden.

Ein anderer Nachteil im Sichtungsprozess könnte das iterative Vorgehen darstellen. Der Fokus lag nie komplett auf einer konkreten Arbeit, sondern auf bestimmte Stellen. Dies hat zwar den Vorteil, dass der Fokus auf die Untersuchung konkreter Forschungsfragen oder auf die Identifikation konkreter Kontexte lag und diese Arbeitsübergreifend verglichen werden konnten, sodass Beziehungen zwischen Kontexte verschiedener Arbeiten bezüglich eines Zweckes besser erkannt werden konnten. Jedoch könnte dadurch das Verständnis einer konkreten Arbeit gemindert werden, sodass implizite Kontexte nicht entdeckt werden konnten.
Tatsächlich wurden drei Arbeiten (a335, a271, a258) nicht tiefer untersucht, da diese in der Identifikation von Kontexten eine hohe Komplexität aufwiesen und nicht ohne weiteres Hintergrundwissen erkannt werden konnten. Alle anderen Arbeiten wurden gründlich untersucht und vor dem Sichten einer Arbeit in der 2. und 3. Iteration

wurden die bereits gelesen Textstellen erneut gesichtet und im Anschluss die bereits identifizierten Kontexte evaluiert. Des Weiteren wurde zu jeder Arbeit eine kurze Zusammenfassung verfasst, die im Kapitel 7 *Anhang* einzusehen sind. Diese Zusammenfassungen sollen nicht nur das Verständnis erhöhen und die erneute Sichtung einer Arbeit erleichtern, sondern auch der Leserin oder dem Leser dieser Masterarbeit einen kurzen Einblick in die Arbeiten geben, die innerhalb dieser Arbeit zitiert wurden.

Eine weitere Schwierigkeit ergibt sich dadurch, dass lediglich eine Person die Arbeiten gesichtet hat, die Kontexte identifiziert und klassifiziert hat und die Analyse durchgeführt hat. Somit besteht zwar ein gewisser Grad an Einheitlichkeit, jedoch könnten Biase entstehen, sodass das Ergebnis dieser Arbeit subjektiv ist.
Die Autorin dieser Masterarbeit hat kaum Forschungserfahrung oder Expertenwissen. Des Weiteren hat sie keine Vorkenntnisse von den technischen Arbeiten der ICSE 2020 oder Erfahrungen mit den Arbeiten, die vor 2020 in den vorherigen ICSE Konferenzen veröffentlicht wurden.
Es könnte sein, dass Kontexte übersehen wurden und unerkannt blieben. Jedoch hängen solche Fehler nicht nur von Erfahrungen und Vorkenntnisse ab, sondern von der Genauigkeit beim Lesen, dass sich nicht genug Zeit für eine Arbeit genommen wird oder eine Routine bzw. ein Trott entwickelt wird. Doch wie bereits im vorherigen Paragraphen erwähnt, hat die Autorin die Arbeiten gründlich in mehreren Iterationen gesichtet und ihre Ergebnisse mehrmals evaluiert und überprüft, sodass eine hohe Genauigkeit im Vorgehen besteht.
Fehler hätten vermieden werden können, wenn mehrere Personen beteiligt wären und die in Kapitel 3 beschriebenen Aktivitäten unabhängig voneinander ohne gegenseitige Beeinflussung für eine zufällig ausgewählte Teilmenge der Arbeiten durchgeführt hätten. Wenn sich dann noch die Teilmengen der untersuchten Arbeiten überschnitten hätten, sodass jede Arbeit mehrmals von unterschiedlichen Personen gesichtet wird, hat es den Vorteil, dass nicht nur mehr Kontexte identifiziert werden könnten, sondern auch, dass die interne Gültigkeit der Arbeit gestärkt wird.
Da diese Masterarbeit selbständig und komplett alleine von der Autorin angefertigt wurde, ist dieser Ansatz nicht möglich. Allerdings wäre es vorstellbar, dass eine Folgearbeit geleistet werden kann, um Ergebnisse dieser Arbeit zu evaluieren und um die interne Gültigkeit dieser Arbeit zu stärken.

Es könnten nicht nur Biase entstehen, sondern auch Verzerrungen. Bei dem Sichten der Arbeiten könnte ein Lerneffekt entstehen, sodass Arbeiten die später in dem Prozess gesichtet, besser untersucht wurden und mehr Kontexte identifiziert wurden. Es wurde versucht diesen Effekt entgegenzuwirken, indem die Arbeiten mehrmals in einen iterativen Prozess gesichtet wurden und die Kontexte samt Textstellen mehrmals evaluiert wurden.

Es ist sehr wahrscheinlich das während des Sichtens der Arbeiten nicht alle Kontexte gefunden wurden und es könnte auch sein, dass Charakteristiken einer Arbeit

unerkannt blieben, falls diese nur implizit angedeutet wurden. Eine maximale bzw. vollständige Abdeckung ist jedoch nicht das Ziel dieser Masterarbeit. Daher wurden auch nicht alle technischen Arbeiten der ICSE 2020 gesichtet. Es ist unklar, wie viele Kontexte unerkannt blieben. Diese Masterarbeit versucht lediglich die ersten Schritte hinsichtlich einer Taxonomie von Software Engineering Kontexten zu machen und nicht einen internationalen Standard zu schaffen. Daher ist weitere Forschung, die auf dieser Masterarbeit aufbaut, nötig.

Durch das mehrmalige Wechseln der Werkzeuge zur Dokumentation, von Microsoft Word zu MaxQDA und zu Microsoft Excel, könnten Fehler bei der Übertragung der Daten gemacht werden. Durch eine gewisse Sorgfalt und mehrmaliges Überprüfen der Daten, sollte die Wahrscheinlichkeit, dass Fehler gemacht wurden, jedoch gering sein.
Auch bei dem Befüllen des Prototyps mit den Daten, könnten Fehler aufgekommen sein und Inkonsistenzen zwischen den Daten in der Microsoft Excel Datei und den Daten in dem Prototypen könnten bestehen, sodass Fehler in der Datenanalyse gemacht wurden. Auch hier wurde durch mehrmaliges Überprüfen versucht die Fehlerwahrscheinlichkeit zu verringern.

5.2 Evaluation der Relevanz

Um herauszufinden, wie relevant eine Taxonomie von Software Engineering Kontexten ist, wurde die Software Engineering Welt in verschiedene Personengruppen segmentiert.

Die Segmentierung eines Marktes ist eine Methode im Marketing, welche genutzt wird um Zielgruppen zu erkennen, um Teilmärkte zu erschließen und um eine Marketing- und Geschäftsstrategie für ein Produkt zu entwickeln. Dabei kann der Markt mittels unterschiedlicher Faktoren eingeteilt werden, wie zum Beispiel anhand des Alters, Einkommens oder des Geschlechts. [61]

Diese Methode wurde sich hier zu Nutze gemacht, um verschiedene Nutzungsszenarien für eine Taxonomie zu erschließen, die Ziele verschiedener Personengruppen aufzudecken und um die Motivation eine Taxonomie zu nutzen von verschiedenen Perspektiven zu evaluieren. Dafür sollen Segmente ausgewählt werden, die durch empirische Methoden genauer betrachtet werden.

5.2.1 Segmentierung

Die Segmentierung des Bereichs Software Engineering wurde anhand der Positionen und Rollen, die eine Person innehaben kann, vollzogen. Es wurde analysiert, wie Personen in Kontakt mit dem Bereich Software Engineering kommen und wie verschieden ihr Werdegang verlaufen kann. Es wurden drei Hauptsegmente mit jeweils drei Untersegmenten abgeleitet, die in der Grafik 5.2 abgebildet sind.

Das erste Segment **Universität** spiegelt die akademische Welt in drei Untersegmenten wider. Es wurde unterschieden zwischen der Rolle des Lernenden, also der Studierenden und der Lehrenden. Personen beider Segmente können Teil des dritten Untersegments Forschungsgruppe sein.

Das zweite Segment **Unternehmen** reflektiert die industrielle Welt in drei Untersegmenten. Dabei wurde analysiert, welche verschiedenen Positionen in einem Unternehmen sind und wie diese aufeinander aufgebaut sind. Diese Positionen wurden anhand der Erfahrungen und Kenntnisse auf zwei Segmente aufgeteilt. Außerdem wurden zusätzlich Positionen von Personen, die nicht selbst Software Engineering anwenden, aber in ihren Arbeitsalltag regelmäßig mit Software Ingenieurinnen und Software Ingenieuren in Kontakt treten, in einem dritten Untersegment gekapselt.

Das dritte Segment **Konferenzen** stellt die verschieden Rollen innerhalb einer Konferenz dar: die Vortragenden, die Arbeiten verfassen, einreichen und vorstellen, die Teilnehmenden, die eine Konferenz besuchen und die Organisatorinnen und Organisatoren einer Konferenz, die die eingereichten Arbeiten begutachten, das Programm der Konferenz gestalten und die Autorinnen und Autoren von Arbeiten als Vortragende einladen.

Diese Segmentierung hat keinen Anspruch auf Vollständigkeit. Es können noch mehrere Segmente bestehen, wie zum Beispiel Institute oder Startups, die sich von Universitäten und Unternehmen in verschiedenen Faktoren unterscheiden. Sicherlich werden ebenfalls nicht alle Werdegänge im Bereich Software Engineering repräsentiert. So kann beispielsweise eine Person einen Quereinstieg wagen oder im Selbststudium Software Engineering erlernen ohne großen akademischen oder industriellen Hintergrund.
Für den Zweck dieser Arbeit, um die Relevanz einer Taxonomie zu evaluieren, werden ausgewählte Untersegmente der bereits erläuterten Hauptsegmente mit verschiedenen Methoden evaluiert.

Ebenfalls ist bei dieser Segmentierung zu beachten, dass diese nicht starr ist. Eine Person kann mehrere Positionen und Rollen innehaben und sich somit in mehreren Segmenten zur selben Zeit verorten. Zum Beispiel kann ein Student als studentischer Mitarbeiter Erfahrungen in einem Unternehmen sammeln oder eine Professorin

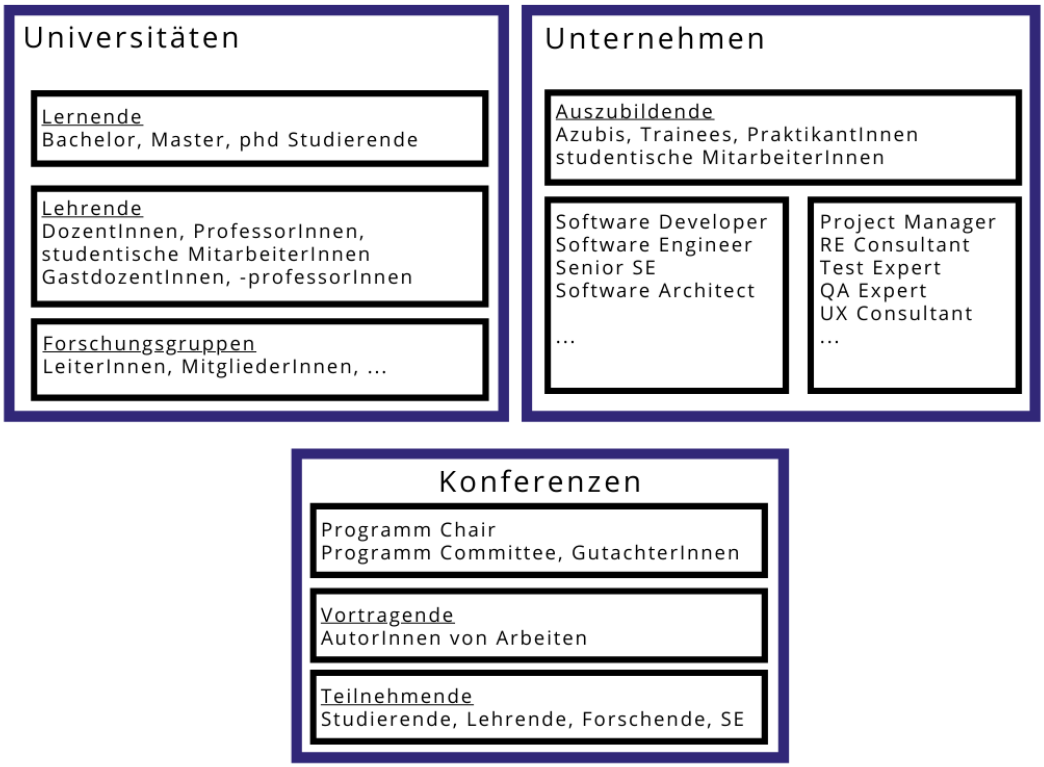


Figure 5.1: Segmentierung des Bereichs Software Engineering

kann Gutachterin in einer Konferenz sein. Ebenfalls können sich die Personen zwischen den Segmenten bewegen und austauschen. Beispielsweise kann der Student sich nach seinem Studium entscheiden, ob er in dem akademischen Umfeld bleibt und eine Forschungslaufbahn anstrebt oder ob er in die Industrie geht und eventuell Software Ingenieur wird. Die Professorin kann ihre Arbeit in Konferenzen oder Unternehmen vorstellen und ihr Wissen mit den anderen Segmenten teilen.

Auch lässt sich nicht generalisieren, dass Universitäten lediglich die akademische Welt widerspiegeln und Unternehmen die industrielle Welt. Unternehmen und Universitäten können Kooperationen haben und im Austausch stehen. Beispielsweise kann ein Student seine Abschlussarbeit innerhalb eines Unternehmens schreiben und eine Software Ingenieurin kann Gastvorträge an der Universität halten. Auch kann eine Universität angewandte Kurse mit industriellen Projekten und Themen anbieten, als auch Praktika in Unternehmen im Studium integrieren. Unternehmen können Forschungsgruppen fördern und Stipendien für Studierende anbieten.

Da in dieser Masterarbeit eine Literaturstudie einer Konferenz durchgeführt wurde und vermutet wird, dass somit dieses Segment Konferenzen am meisten von den Ergebnissen profitieren würde, wurde der Fokus dieser Relevanzanalyse auf das dritte Segment Konferenzen gelegt.

Dazu wurden semi-strukturierte Interviews mit Gutachterinnen und Gutachter der ISCE 2020 durchgeführt, welche in der Sektion 5.2.2 erläutert werden.

Vermutung: Es wurde sich für das Segment Gutachterinnen und Gutachter entschieden, da vermutet wird, dass diese durch ihre Erfahrungen auch andere Segmente abdecken und unterschiedlichen Perspektiven einnehmen können. Es ist wahrscheinlich, dass Gutachterinnen und Gutachter auch die Rollen der Vortragenden oder Teilnehmenden einer Konferenz eingenommen haben und sich somit auch in den anderen Untersegementen des Segmentes Konferenz bewegt haben. Außerdem wird vermutet, dass Gutachterinnen und Gutachter zusätzlich forschen und lehren und sich neben dem Segment Konferenz überwiegend in dem Segment Universität bewegen.

Die anderen Segmente sollen bei dieser Relevanzanalyse jedoch nicht außer Acht gelassen werden.
Das Segment Unternehmen soll durch Usability Tests abgedeckt werden, welche mit Software Engineers und UX Expertinnen und Experten durchgeführt werden und welche in der Sektion 5.3 zur Evaluierung des Prototyps *TSEK-Analyzer* beschrieben werden. Außerdem wurden Autorinnen und Autoren kontaktiert, um an einer Umfrage teilzunehmen, welche in Sektion 5.4 zur *Evaluation der Taxonomie* beschrieben wird. Weitere Segmente werden in Sektion 5.2.3 im Anschluss der folgenden Sektion analysiert.

5.2.2 Semi-Strukturierte Interviews mit Gutachterinnen und Gutachtern

Es wurden alle 37 Gutachterinnnen und Gutachter angeschrieben und zu einem Interview eingeladen, die das Programm Board der ICSE 2020 bilden [16]. Das Anschreiben dazu, so wie die Fragen des semi-strukturierten Interviews befinden sich im Anhang dieser Arbeit in Sektion 7.4.
Von den 37 Gutachterinnen und Gutachter haben 9 geantwortet. Dabei haben 5 aus Zeit- oder Kapazitätsgründen abgesagt. Eine Person hat die Fragen per Mail beantwortet und 3 Interviews wurden virtuell durchgeführt.

Insgesamt wurden 16 Fragen vorbereitet, welche in vier Kategorien unterteilt werden können: Verständnis des Einreicheprozess einer Arbeit, Verständnis des Gutachtungsprozesses, Segmentierung und Austausch zwischen den Segmenten und Taxonomie.
Die Interviews wurden semi-strukturiert durchgeführt und waren auf eine halbe Stunde begrenzt. Dabei wurden nicht alle Fragen gestellt. Die vorbereiteten Fragen dienten als Unterstützung um das Interview zu führen. Verständnisfragen, die bereits

[16] https://2020.icse-conferences.org/committee/icse-2020-technical-papers-program-board

durch ein vorheriges Interview ausreichend beantwortet wurden, wurden nicht erneut in einem weiteren Interview gestellt.

Aus den Interviews konnten mehrere Nutzungsszenarien der Taxonomie für verschiedene Segmente bzw. Personengruppen abgeleitet werden. Diese werden im Folgendem vorgestellt und evaluiert.

Nutzungsszenario: Bewerten einer Arbeit Segment: Gutachterinnen und Gutachter
Auf die Frage, ob die Taxonomie bei dem Begutachten einer Arbeit unterstützend wirken könnte, waren die Meinungen verschiedenen.
Ein Befragter meinte, es könnte helfen, eine Arbeit leichter einzuschätzen und auch um besser Feedback zu geben, wenn es einen Standard gäbe. Eine Taxonomie könnte also die Kommunikation zwischen Gutachterinnen, Gutachtern, Autorinnen und Autoren fördern, sowie die Verortung der Forschung.
Ein weiterer Befragter hat auf die Klassifikation von ACM und die CCS Konzepte verwiesen [17], und ergänzt, dass es bereits viele verschiedene Versionen von Klassifikationen in dem Bereich Software Engineering gibt und gefragt, wieso eine weitere im Rahmen dieser Arbeit entwickelt werden soll. Diese Frage wird im weiteren Verlauf dieser Arbeit in Sektion 5.4 zur Evaluation der Taxonomie beantwortet.
Eine andere Befragte, hat geäußert, dass eine Taxonomie die Art von Paper auf der ICSE einschränken könnte und dass es bereits weniger Flexibilität gibt, als früher. Darauf wird näher bei der Erläuterung des zweiten Nutzungsszenario *Verfassen und Einreichen einer Arbeit* eingegangen. Es waren sich alle Befragten einig, dass die Art von Paper einen Einfluss auf die Bewertungskriterien hat.

Aus den Interviews lässt sich ableiten, dass jede Gutachterin, jeder Gutachter eine andere Strategie hat ein Arbeit zu begutachten. Beispielsweise bevorzugt es ein Gutachter das Ende einer Arbeit zuerst zu lesen und fängt von Hinten an. Ein anderer fängt von Vorne an und wenn er nicht zurecht kommt, springt er zu anderen Sektionen: Zu den Referenzen, um den Kontext der Arbeit zu analysieren, zu den Schlussfolgerungen, um die Ziele und die Ergebnisse der Arbeit zu verstehen oder zu dem Ansatz, um zu verstehen, wie die Autorinnen und Autoren ihre Ziele erreichen.
Auch das Gutachtens wird von den Gutachterinnen und Gutachtern zu unterschiedlichen Zeitpunkten verfasst. Ein Befragter schreibt zum Beispiel erst eine Zusammenfassung von einer Arbeit, bevor er das Gutachten verfasst. Ein anderer Befragter liest sich die Arbeit erneut durch um im Anschluss ein Gutachten zu verfassen.

Schlussfolgerung: Man kann schlussfolgern, dass es unterschiedliche Lese- und Begutachtungsstrategien gibt, abhängig von den eigenen Präferenzen. Es wird vermutet, dass lediglich eine Teilgruppe der Gutachterinnen und Gutachter eine Taxonomie unterstützend in ihre Arbeit benutzen würde, wenn diese nicht als offizielles

[17] https://dl.acm.org/ccs

Werkzeug zur Bewertung eingeführt wird.

Nutzungsszenario: Verfassen und Einreichen einer Arbeit Segment: Autorinnen und Autoren
Das zweite Nutzungsszenario, dass Autorinnen und Autoren bei dem Verfassen und Einreichen ihrer Arbeiten unterstützt werden, bezieht sich zwar nicht auf das Segment der Rolle der befragten Gutachterinnen und Gutachter. Jedoch haben alle Befragten selbst Paper verfasst und veröffentlicht, sodass sie diese Rolle gut repräsentieren können.

Ein Befragter hat hervorgehoben, dass eine Taxonomie ein Standard sein könnte, an denen sich Autorinnen und Autoren orientieren können. Die Dimensionen der Generalisierbarkeit zu kennen, wäre ein erster Schritt. Es hätte nicht nur den Vorteil, dass die Autorinnen und Autoren sich den Dimensionen bewusst sind, sondern dass diese auch die gleiche Sprache verwenden.

Im Gegensatz dazu steht wieder die mögliche Einschränkung der Flexibilität, die bereits im ersten Nutzungsszenario angesprochen wurde. Ein Standard kann Arbeiten einschränken, die diesen nicht befolgen, sodass diese unter Umständen nicht auf einer Konferenz angenommen werden. Dabei können Standards sowohl als Orientierung genutzt werden, um eine Arbeit zu verfassen, aber auch um eine Qualität von Arbeiten zu sichern. So können die Autoren und Autorinnen die Bewertungskriterien der ICSE zur Orientierung nutzen, um die Qualität ihrer Arbeit zu erhöhen. Dabei wird auch die Wahrscheinlichkeit, dass ihre Arbeit akzeptiert wird, erhöht. Zusätzlich zu den Bewertungs- bzw. Akzeptanzkriterien von Arbeiten haben viele Magazine oder Konferenzen auch Standards bzw. Anforderungen zu der stilistischen Struktur einer Arbeit, um unter Anderem die Qualität von Arbeiten zu sichern.
Die befragte Person, die die Befürchtung äußerte, dass die Arbeiten eingeschränkt werden könnten, war der Meinung, dass eine gute Arbeit den Umfang und Einfluss erklärt und dass es generell wünschenswert ist, dass in einer Arbeit deren Kontext klar umrissen wird.

Schlussfolgerung : Daraus lässt sich schlussfolgern, dass die explizite Beschreibung des Kontextes eine Qualitätsfaktor sein kann und dass es zu einer höheren Qualität kommen könnte, da die Autorinnen und Autoren aufgefordert werden ihren Kontext zu beschreiben.

Jedoch haben andere Befragte geäußert, dass wenn der Kontext einer Arbeit explizit beschrieben wird, insbesondere bezüglich der Verallgemeinerbarkeit, dies auch bewiesen werden muss.
Bereits bei der Analyse zur Verallgemeinerbarkeit in Sektion 4.2 fällt auf, dass viele

Autorinnen und Autoren nur vage zur Verallgemeinerbarkeit ihrer Ergebnisse Stellung nehmen. Durch die Implementierung eines Standards könnten, deren Arbeit eingeschränkt werden und die Autoren wären gezwungen Stellung zu nehmen.
Ein befragter Gutachter ist der Meinung, dass Verallgemeinerbarkeit subjektiv ist, da es keine harten Fakten sind, sondern Plausibilitätsargumente. Die Verallgemeinerbarkeit von Ergebnissen ist schwierig zu beweisen, und je konkreter und klarer die Autorinnen und Autoren Stellung nehmen, umso größer ist die Beweislast.
Wenn die Stellungnahme zur Verallgemeinerbarkeit zu weit gefasst wird oder nicht gut genug begründet wird, kann das Auswirkungen, auf den Gutachtungsprozess haben. Gutachterinnen und Gutachter können nicht einverstanden sein, die Arbeit kritisieren und diese somit möglicherweise nicht auf der ICSE zulassen, als wenn die Arbeit lediglich eine vague Stellungnahme äußert.

Ein Gutachter hat erwähnt, dass eine Methodik, die mehr als nur auf Plausibilität beruht, um die Verallgemeinerbarkeit einer Arbeit einzuschätzen, fehlt.

Schlussfolgerung : Es kann geschlussfolgert werden, dass eine Taxonomie nicht als Standard zur Qualität eingeführt werden kann und auch nicht von Autorinnen und Autoren angenommen wird, solange es Schwierigkeiten gibt den Kontext einer Arbeit zu beschreiben und zu beweisen. Zuerst sollte eine Methodik entwickelt werden, die Autorinnen und Autoren unterstützt die Verallgemeinerbarkeit und somit den größeren Kontext ihrer Arbeiten einzuschätzen und vor allem begründen zu können.

Nutzungsszenario: Verortung von Arbeiten und Beweisführung Segment: Autorinnen und Autoren
Die Taxonomie könnte unterstützend wirken und den Autorinnen und Autoren möglichen Dimensionen der Kontexte aufzeigen, sodass eine Verortung der eigenen Arbeit erleichtert wird.
In dem implementierten Werkzeug *TSEK-Analyzer* wird visualisiert, welche Arbeiten der hier durchgeführten Literaturstudie der ICSE 2020, sich in welchen Kontexten bewegen. Autorinnen und Autoren, die sich auf diese Arbeiten beziehen, können dadurch Schlussfolgerungen auf ihren eigenen Kontext ziehen, ihre Arbeit besser einschätzen und möglicherweise neue Einsichten gewinnen.

Des Weiteren können durch eine Taxonomie die Arbeiten anderer Autorinnen und Autoren besser verortet werden. Ist der Kontext einer Arbeit explizit dargestellt, können andere Autorinnen und Autoren leichter einschätzen, ob die Ergebnisse dieser Arbeit für die eigene Arbeit relevant sind. Ist der Kontext einer Arbeit nicht beschrieben, können lediglich Annahmen über den Kontext getroffen werden und Arbeiten, die auf Ergebnisse anderer Arbeiten basieren, könnten fehlgeleitet sein, sich fälschlicherweise auf Ergebnisse anderer Arbeiten stützen und diese fälschlicherweise zur Beweisführung nutzen.

Schlussfolgerung : Eine Taxonomie könnte helfen, dass die Forschung von Software Engineering weniger fehleranfällig wird und eine Terminologie zur Kontextbeschreibung schaffen. Dabei stellt die Taxonomie jedoch lediglich ein Werkzeug zum Einschätzen des Kontextes dar, und nicht zum Einschätzen oder zum Beweisen der Verallgemeinerbarkeit einer Arbeit.

Nutzungsszenario: Zuweisungsprozess der Gutachterinnen und Gutachter für eine Arbeit Segment: Gutachterinnen und Gutachter
Die Taxonomie kann jedoch nicht nur den Autorinnen und Autoren bei der Verortung ihrer Arbeiten helfen, aber auch den Gutachterinnen und Gutachtern, um das Zuweisen einer Arbeit zum Begutachten zu erleichtern.
Beim Einreichen einer Arbeiten müssen die Autorinnen und Autoren bereits 2-3 Themen angeben, damit die Gutachterinnen und Gutachter, die dessen Arbeit bewertet, besser zugewiesen werden können. Es wäre vorstellbar, dass die Autorinnen und Autoren im Einreicheprozess ihren eigenen Kontext in einer Taxonomie verorten könnten, um anhand dessen Selbsteinschätzung die bestgeeignetsten Gutachterinnen und Gutachter zu finden, die sich in diesem Kontext auskennen und entsprechende Expertise aufweisen.
Momentan wird diese Zuweisung durch ein zeitaufwändigen Prozess durchgeführt. Gutachterinnen und Gutachter stehen die Abstracts der Paper zur Verfügung, anhand dessen sie Paper auswählen können, die sie begutachten möchten. Die Gutachterinnen und Gutachter müssen selbstständig einschätzen, zu welchen Paper sie ausreichende Expertise haben, um das Paper begutachten zu können. Dabei spielen auch Präferenzen zu bestimmten Papertypen und Themen eine Rolle.
Auf der ICSE 2020 wurden 617 Arbeiten eingereicht, die von mindestens 3 der 37 Gutachterinnen und Gutachter bewertet wurden. Um die Arbeiten zum Gutachten auszuwählen, ist es sehr zeitaufwändig alle Abstracts der 617 eingereichten Arbeiten zu lesen, sodass ein Optimierungsbedarf in diesen Prozess besteht, damit die Gutachterinnen und Gutachter, die in einem Themengebiet Expertinnen und Experten sind, auch die passenden Arbeiten begutachten.
Wenn die Autorinnen und Autoren beim Einreichen ihrer Arbeit, den Kontext ihrer Arbeit definieren und in einer Taxonomie einfügen, können Gutachterinnen und Gutachter sich in dieser Taxonomie positionieren und die für sie interessanten Arbeiten schneller finden und somit Arbeiten bewerten, die in ihren Feld liegen.

Schlussfolgerung : Eine Taxonomie könnte den Zuweisungsprozess von Arbeiten und Gutachterinnen und Gutachtern optimieren. Dazu müsste es als offizielles Werkzeug einer Konferenz eingeführt werden und die Möglichkeit bieten, dass Autorinnen und Autoren die Taxonomie verändern und erweitern können, sodass der Kontext ihrer Arbeit abgebildet wird.

5.2.3 Analyse weiterer Segmente

Im Folgendem wird evaluiert inwieweit weitere Personengruppen und Segmente von einer Taxonomie von Software Engineering Kontexten profitieren könnten.

In den vorherigen Sektionen dieser Arbeit wurde bereits die Personengruppen Gutachterinnen und Gutachter und Verfasserinnen und Verfasser von Arbeiten des Segmentes Konferenz betrachtet. Die dritte Personengruppe in diesem Segment sind die Teilnehmenden der Konferenz. Wie bereits in Kapitel 2.2 erwähnt, richtet sich die ICSE an Forscherinnen und Forscher, Praktizierende und Lehrende, welche in den Segmenten Universität und Unternehmen abgebildet sind.

Nutzungsszenario: Auswahl der Vorträge Segment: Teilnehmende an Konferenzen
In der Rolle als Teilnehmende an einer Konferenz könnten die Teilnehmenden leichter Vorträge finden, die für sie relevant sind, und somit leichter entscheiden, welche Vorträge sie besuchen möchten, wenn die Kontexte der Forschungsbeiträge in einer Taxonomie abgebildet sind,
Anstatt eine Liste von vorgestellten Arbeiten [18] durchzugehen, könnten sie nach konkreten Kontexten suchen, die in ihrem Interessengebiet liegen oder mit den Kontext ihrer eigenen Arbeit bzw. Forschung verknüpft sind. Anhand dieser konkreten Kontexte könnten entsprechende Forschungsbeiträge gefunden werden, welche auf der Konferenz vorgestellt werden.
Auch könnten die Teilnehmenden nachdem sie einen Vortrag besucht haben, den Kontext dieses Vortrags in der Taxonomie finden und Vorträge, die im selben Kontext liegen, suchen und somit die Forschung besser verorten.

Nutzungsszenario: Recherche Segment: Universität, Autorinnen und Autoren
Die Verortung der Forschung ist nicht nur für Teilnehmende einer Konferenz interessant, sondern auch für andere Segmente.
Für Autorinnen und Autoren und Personengruppen in dem Segment Universität könnte dieses Nutzungsszenario, besonders in der Recherche, von Nutzen sein. Verwandte Arbeiten könnten durch die Taxonomie über dessen Kontexte verknüpft werden, sodass die Literatursuche und -auswahl erleichtert wird.
Forschungsgruppen, die ihre Beiträge veröffentlichen und somit auch in der Rolle Verfasserinnen und Verfasser von Arbeiten unterwegs sind, könnten ihr eigene Forschung besser einordnen, verwandte Arbeiten zu ihrer eigenen Arbeit finden und interessante Arbeiten zum Vorstellen in Wissenstransfer Formaten der Forschungsgruppe finden.
Lehrende könnten den Kontext ihrer Vorlesungen besser aufzeigen und vermitteln und für ihre Vorlesungen relevante und aktuelle Literatur finden.
Studierende, die meist wenig Erfahrung in der Recherche haben, könnten sich einen

[18] Programm der ICSE 2020 https://conf.researchr.org/track/icse-2020/icse-2020-papers?program

Überblick verschaffen, ein besseres Verständnis der aktuellen Forschung erlangen und ihre Interessengebiete einordnen.

Nutzungsszenario: Themenwahl für neue Forschungsbeiträge Segment: Universität
Eine Taxonomie kann nicht nur bei der Literaturauswahl während der Recherche helfen, sondern auch bei der Themensuche und -auswahl. Vielen Studierenden fällt es schwer ein geeignetes Thema für ihre Abschlussarbeiten zu finden. Mit einer Taxonomie können sie ihre Interessen mit den Kontexten und somit mit den aktuellen Forschungsbeiträgen abgleichen und Ideen für das Thema einer eigenen Arbeit entwickeln.
Außerdem könnten Lücken und Trends besser erkannt werden, sodass sich das Thema und der Kontext einer Forschungsarbeit leichter finden lässt. Dafür könnte die Taxonomie so angepasst werden, dass auch die Charakterisierung der Arbeiten involviert wird. Dadurch kann eingesehen werden, welche Vorgehensweisens innerhalb welcher Kontexte bereits genutzt wurden und bei Forschungsbedarf kann der selbe Kontext mit anderen Methoden untersucht werden.

Nutzungsszenario: Weiterbildung Segment: Unternehmen
Während des Interviews wurden die Befragten auch bezüglich der Segmentierung und dem Austausch zwischen den Segmenten befragt. Zwei Befragte waren der Meinung, dass der Austausch zwischen Industrie und Akademie nicht gut sei. Es gebe wenig Austausch und wenig gemeinsame Projekte. Fast alle Konferenzen seien akademisch und die Industrie sei nicht in der Universität vertreten. Beide Befragten verweisen auf andere Felder der Informatik, wie Machine Learning, Cloud Computing und Computergrafik, in denen mehr zwischen Universitäten und Unternehmen interagiert wird, zum Beispiel indem Firmen Labs an den Universitäten gründen oder mehr auf Konferenzen vertreten sind. Ein Befragter ist der Meinung, dass persönliche Verbindungen eine große Rolle spielen.
Was die Ursachen für den anscheinend mangelnden Austausch zwischen den Segmenten sind, müsste in weiterer Forschung geklärt werden. Eventuell könnte die Taxonomie dazu beitragen eine gemeinsame Terminologie zu finden. Die Taxonomie könnte die Verortung der Forschung fördern und somit auch die Transparenz der Forschung. Außerdem könnte auch der Wissenstransfer zwischen Forschung und Industrie unterstützt werden, sowie die akademische Weiterbildung von Software Ingenieurinnen und Ingenieure nach ihrem Studium.

Annahme: Es wird angenommen, dass aus dem Segment Unternehmen die Personen aus dem Untersegment Praktizierende den größten Nutzen an einer Taxonomie hätten. Das Segment Auszubildende und das Segment mit den Rollen, die im Arbeitsalltag im Kontakt mit Praktizierenden stehen, könnten eine Taxonomie jedoch nutzen, um sich einen Überblick über den aktuellen Stand der Forschung zu verschaffen.

Um mehr Einblicke in die Relevanz der Taxonomie für Unternehmen zu erlangen und um herauszufinden inwiefern Praktizierende zur Forschung in Kontakt stehen, wurden Software Ingenieurinnen und Ingenieure semi-strukturiert interviewt. Die Beiträge des Interviews werden in der nächsten Sektion 5.3.3 erläutert und dieses Nutzungsszenario wird erneut in dieser Sektion aufgegriffen und evaluiert.

5.3 Evaluation des Prototyps: TSEK-Analyzer

Der Prototyp TSEK-Analyzer, welcher die Ergebnisse dieser Masterarbeit, wie die Taxonomie, darstellt, wurde durch Usability Tests evaluiert.
Dabei wurden verschiedene Ziele verfolgt. Das Ziel eines Usability Tests ist es nicht nur das Ziel, die Benutzerfreundlichkeit eines Produkts zu testen, sondern ebenfalls zu testen, ob bestimmte Aufgaben effektiv und effizient von Erstnutzer gelöst werden können. Mittels geeigneter Testaufgaben kann frühzeitig erkannt werden, ob Unstimmigkeiten im Design oder Fehler in der Umsetzung von Funktionalitäten vorhanden sind.
Jedoch sollen durch diese Usability Tests vor Allem verschiedene Perspektiven betrachtet werden und ein Verständnis von potentiellen Nutzerinnen und Nutzer erlangt werden. In Kapitel 4 in Sektion 4.5 wurde angenommen, dass die Visualisierung der Taxonomie, eine interaktive Darstellung der Ergebnisse dieser Arbeit und weitere Nutzungsszenarien einen potentiellen Nutzen für andere Personengruppen hat. Diese Annahme soll nicht nur die vorangegangene Evaluierung der Relevanz überprüft werden, aber auch durch die Usability Tests.

Dabei soll jedoch lediglich ein MVP Prototyp implementiert werden, mit den nötigsten Funktionalitäten um unterstützend bei der Verfassung dieser Masterarbeit zu wirken. Ein ausgereifter Prototyp, der Funktionalitäten für verschiedene Nutzungsszenarien abdeckt und an verschiedene Personengruppen angepasst ist, ist nicht Ziel dieser Arbeit. Allerdings sollen weitere potentielle Nutzungsszenarien aufgedeckt werden, um die Relevanz der Taxonomie zu evaluieren, um potentielle Erweiterungsmöglichkeiten des Prototypens zu bestimmen und um nächste Schritte zu einer Taxonomie von Software Engineering Kontexten ausmachen zu können.

5.3.1 SetUp

Um Personengruppen innerhalb der industriellen Welt zu evaluieren, wurde ein Usability Test in der Industrie durchgeführt mit Teilnehmenden, die beruflich bei dem Beratungs- und IT-Dienstleistungsunternehmen *adesso SE*[19] tätig sind. Da keine

[19] https://www.adesso.de/

persönlichen Beziehungen zu den Teilnehmenden persönlich bestehen, ist eine Verzerrung der Ergebnisse ausgeschlossen.
Innerhalb adesso SE wurde ein Usability Test Event von einen Software Engineer und einen Berater für Requirements Engineering und User Experience organisiert und moderiert. Beide Moderatoren sind bereits mehrere Jahre beruflich tätig und verfügen über Erfahrungen in der Organisation, Planung und Umsetzung von Usability Tests. An dem Event wurden fünf Projekte von acht Teilnehmenden getestet.

Aufgrund von Covid Einschränkungen wurden die Usability Tests remote umgesetzt. Dazu wurde die Plattform *Microsoft Teams* [20] verwendet, welche Video-Besprechungen innerhalb *Breakout-Rooms* ermöglicht und somit auch die Aufteilung der Teilnehmenden in seperaten Besprechungsräumen. Für die fünf zu testenden Projekten gab es je einen Besprechungsraum. TSEK-Analyzer konnte mit sechs der acht Teilnehmenden getestet werden.
Die Zeit zur Durchführung eines Tests wurde auf 12 Minuten begrenzt und nach Ablauf der Zeit haben die Moderatoren des Events getaktet die Breakout-Rooms gewechselt, sodass ein effizienter Wechsel zwischen den Teilnehmenden und Testleitenden möglich war und viele verschiedene Perspektiven von unterschiedlichen Teilnehmenden gesammelt werden konnten.

Um Zeit zu sparen, haben die Moderatoren die Teilnehmenden vor der Durchführung der Usability Tests darauf vorbereitet, wie das Event ablaufen wird, erklärt, wie ein Usability Test generell abläuft und erläutert, wie die *Think Aloud* Methode funktioniert. Außerdem wurde betont, dass nicht die Teilnehmenden getestet werden, sondern die Prototypen. Fehler sind erwünscht, sodass die Testleitenden Fehler und Inkonsistenten in ihren Projekten früh erkennen können und den Prototypen frühzeitig im Projektverlauf verbessern können und an den Nutzern anpassen können.
Des Weiteren haben sich vor der Durchführung der Usability Tests alle Teilnehmende kurz vorgestellt und ihre Erwartungen geteilt. Die Testleitenden haben zusätzlich kurz den Hintergrund zu ihren Prototypen präsentiert.

Da der Prototyp zu dem Zeitpunkt der Durchführung nicht deployed war, konnten die Teilnehmenden den Prototypen nicht selbstständig oder lokal auf ihren eigenen Computer erkunden. Der Bildschirm wurde geteilt und der Teilnehmende musste die Moderatorin anleiten, indem laut gedacht wird, was einem besonders auffällt und was man als nächstes tun würde. Dies hatte den Vorteil, dass durchgängig die *Think Aloud* Methode angewandt wurde.
Wegen dieser Einschränkung lag der Fokus jedoch weniger auf die Interaktion zwischen den einzelnen Komponenten des Prototyps, und mehr auf die Visualisierung der Informationen, da auch zu dem Zeitpunkt noch nicht alle Funktionalitäten und Interaktionen zwischen den Komponenten umgesetzt waren. Des Weiteren waren auch noch nicht alle Daten zu allen gesichteten Arbeiten im Prototypen bereitgestellt.

[20] https://www.microsoft.com/de-de/microsoft-teams/group-chat-software

Als Vorbereitung und als Unterstützung für die Usability Tests wurde vorab ein Testbogen erstellt, welcher in Kapitel 7 zum Anhang in Sektion 7.3 bereitgestellt wurde. Der Testbogen enthält Felder zur Datenerhebung zur Charakterisierung der Person, Aufgabenstellungen und Fragestellungen für ein Feedback bezüglich der Benutzerfreundlichkeit des Prototyps und für ein semi-strukturiertes Interview. Dieser Bogen wurde ausgedruckt und als ein schriftliches Protokoll genutzt.
Während des Tests wurden zwei Rollen gleichzeitig eingenommen, die Rolle der Moderatorin und die Rolle der Protokollantin. Dies stellte zwar eine Herausforderung dar. Jedoch war dies keine ungewohnte Situation, da die Autorin dieser Arbeit bereits Erfahrungen im akademischen, sowie im industriellen Umfeld, mit der Planung und Umsetzung von Usability Tests sammeln konnte.

Zu Beginn jedes Tests, hat die Verfasserin, die den Test moderiert und protokolliert hat, sich vorgestellt und ihre Masterarbeit kurz umrissen, bevor sie Daten zur Testperson erhoben hat. Des Weiteren wurde, bevor die Aufgaben gestellt wurden, der Prototyp kurz erneut vorgestellt und erklärt, dass der Prototyp sich nicht in einem endgültigen Status befindet.
Bei den erhobenen Daten handelt es sich dabei um folgende personen-bezogenen Informationen: Name, Geschlecht, akademischer Abschluss, Abschlussjahr, Position und Berufserfahrung in Jahren. Um Zeit zu sparen, wurden nicht alle Informationen für jede Testperson abgefragt, sondern nur die Nötigsten, um während des Test abschätzen zu können zur welcher Personengruppe die Teilnehmende gehören. Die restlichen Informationen wurden sich im Nachhinein beschafft durch die Nutzung eines internen Werkzeug zur Profilsuche der Mitarbeiterinnen und Mitarbeiter bei *adesso SE*.
Die Teilnehmenden können abhängig von ihrer Rolle und Position bei *adesso SE* in zwei Personengruppen abhängig von ihrem Berufsfeld zugeordnet werden: Requirements Engineering oder Software Engineering. Es haben vier Personen mit dem Berufsfeld Requirements Engineering teilgenommen und zwei Personen mit dem Berufsfeld Software Engineering.

Da nicht alle Teilnehmende Domänenwissen im Software Engineering hatten, lag der Fokus des Test bei diesen Personen mehr auf die Benutzerfreundlichkeit und *User Experience* des Prototyps. Bei Personen mit mehr Domänenwissen wurden mehr Fragen bezüglich der visualisierten Informationen gestellt.
Außerdem wurden die Tests von der Moderatorin so gelenkt, dass die Teilnehmenden, die die selben Usability Probleme erkennen, nicht wiederholend dieses Problem ausführlich erläutern, damit innerhalb der begrenzten Zeit so viele Probleme wie möglich erkannt werden können und so viele Einsichten wie möglich erlangt werden können.

Während des Tests wurden zwei Aufgaben gestellt. Die erste Aufgabe beschäftigt sich mit dem Nutzungsszenario Recherche, welches zwar kein mit Absicht implementiertes Nutzungsszenario des Prototyps ist, jedoch wird vermutet, dass andere Personengruppen, die nicht diese Masterarbeit verfassen, dieses Nutzungsszenario

als nützlich erachten. In der ersten Aufgabe wurde die Testperson zunächst darum gebeten nach einen aktuellen Beitrag zu einem Thema ihrer Wahl im Bereich Software Engineering zu recherchieren.
Die zweite Aufgabe handelt von der Datenanalyse und -visualisierung. Es ging darum zu beschreiben, welche Informationen der Testperson auffällt, sobald sie eine Arbeit näher betrachtet und welche Informationen sie als interessant betrachtet.
Nach dem Beenden der Aufgaben wurde nach einem Feedback gefragt, welche Verbesserungsmöglichkeiten bestehen, was gut oder schlecht war hinsichtlich der Usability und der Visualisierung der Daten.

5.3.2 Ergebnisse der Usability Tests

Es wurden vier Anforderungen bzw. Erweiterungen aus den erkannten Problemen und erlangten Einsichten entwickelt, die in der Tabelle 5.1 aufgelistet sind und im Folgendem erläutert werden. Im Folgendem werden diese Anforderungen erläutert und evaluiert, ob diese als Erweiterung in den Prototypen aufgenommen werden.
Auf kleine Erweiterungen für kleinere Benutzerfreundlichkeit Probleme bezüglich der Schriftgröße, Nutzung von Icons und Beschriftungen, die aufgedeckt wurden, wird im Folgendem nicht näher eingegangen, da diese keinen großen Aufwand in der Implementierung darstellen.

#	Anforderungen	Testpersonen	Nutzungsszenario
1	Alles-Suche	T2, T3, T5	Recherche
2	Komponenten-übergreifende Interaktion	T1, T5	Datenanalyse
3	Konfiguration der Anzeige	T1, T5	Datenvisualisierung
4	Anzeige anderer Informationen	T4, T5	Datenvisualisierung

Table 5.1: Erkannte Anforderungen aus den Usability Tests

Anforderung: Alles-Suche
Drei der Teilnehmenden haben sich für die Recherche eine generelle Suche gewünscht, anstatt einen spezialisierten Filter. Dies wäre zum einen mehr komfortabel, da es weniger kompliziert ist, und zu anderen entspreche es mehr die Erwartungen nach der Eingabe eines Suchbegriffes per Enter direkt alles suchen zu können, anstatt ein Filterkriterium auszuwählen.
Der implementierte Filter könnte hinter einen Link zur ”erweiterten Suche” versteckt werden und lediglich ein Suchschlitz, der über alle Attribute bezüglich einer Arbeit sucht, könnte angeboten werden.
Da die primäre Nutzerin, die Verfasserin dieser Arbeit, nach konkreten Kriterien die Arbeiten filtert, um ihre Annahmen zu bestätigen und Schlussfolgerungen ziehen zu

können, ist für sie eine generelle Suche weniger sinnvoll. Jedoch kann der Prototyp in der Zukunft um diese Funktionalität erweitert werden, um die Recherche und Datenanalyse für andere Personengruppen zu erleichtern. Jedoch soll der Prototyp nicht Suchmaschinen wie *Google Scholar* ersetzen um empirische Arbeiten zu recherchieren, sondern konkret die Arbeiten der ICSE 2020 aufbereiten, Ergebnisse der gemachten Analysen darstellen und Zusammenhänge zwischen Arbeiten, Kontexte und anderen Daten erkennbar machen.

Anforderung: Komponenten-übergreifende Interaktion
Eine Teilnehmerin hat erwartet, dass ein User-Aktion innerhalb der Filter- oder Übersichtskomponente ein Event in der Graphkomponente ausgelöst. Da bei der Durchführung der Usability Tests lediglich ein Zwischenstand des Prototyps gezeigt wurde, war diese Anforderung, die bereits angedacht war, noch nicht implementiert und somit wurde diese Erwartung nicht erfüllt.
Insbesondere die Filterkomponente soll Einfluss auf die Graphkomponente nehmen und die Taxonomie der Kontexte anpassen. Des Weiteren soll in der Übersichtskomponente durch einen Button, der Viewport der Graphkomponente auf die Kontexte einer konkreten Arbeit fokussiert werden. Diese Interaktion wurde auch durch einen anderen Teilnehmer vorgeschlagen.
Bei weiteren Usability Tests wurde darauf hingewiesen, dass diese Interaktion noch nicht implementiert ist, um andere Erkenntnisse zu erlangen.

Anforderung: Konfiguration der Anzeige
Einem Teilnehmer schien die Oberfläche durch die Menge an Informationen zu überladen. Er hätte sich gewünscht, dass die Taxonomie erstmal versteckt sei und dass es zwei verschiedene Ansichten gibt, zwischen denen gewechselt werden kann. Daraufhin wurde auf die Funktionalität hingewiesen, dass die Filterkomponente versteckt werden kann, die Graphkomponente auf die volle Fensterweite gestreckt werden kann und auch die Anzeige verschiedene Informationen innerhalb der Übersicht einer konkreten Arbeit konfiguriert werden konnte (mittels den implementierten VisibilityService).
Das wurde als ein guter erster Ansatz angenommen, insbesondere damit Screenshots mit lediglich den relevanten Informationen gemacht werden können. Dieses Nutzungsszenario, dass Nutzerinnen und Nutzer Informationen teilen möchten, wurde erst durch diesen Usability Test erkannt. Da auch die Autorin ihre Ergebnisse teilen möchte, wurde die Idee entwickelt, dass die Konfiguration der Anzeige durch die Url kontrollierbar sein sollte. Dadurch kann sie durch das Teilen einer URL die relevanten Informationen zu den Ergebnissen konkreter Analysen teilen und präsentieren. Des Weiteren soll es auch möglich sein die Übersichtskomponente auf voller Breite ansehen zu können und den Graphen zu verstecken.

Anforderung: Anzeige anderer Informationen
Die letzte Anforderung betrifft die Anzeige von konkreten Informationen in der Übersichtskomponente. Es wurde angemerkt, dass die Namen der Autorinnen und Autoren einer Arbeit nicht direkt einsehbar sind. Diese sind hinter einem Icon zu finden, nachdem auf eine konkrete Arbeit geklickt wurde.
Des Weiteren wurden sich die Information zum Erscheinungsdatum, eine Kurzzusammenfassung und Tags über das Themengebiet einer Arbeit gewünscht. Auch wurde angemerkt, dass die Beziehungen zwischen den Kontexten innerhalb einer Arbeit in der Übersichtskomponente besser visualisiert werden solle.
All diese Informationen könnten für bestimmte Personengruppen nützlich, jedoch sind sie nicht unbedingt relevant für die primäre Nutzerin und für den MVP, der innerhalb dieser Masterarbeit umgesetzt wird.

Letztendlich muss der Nutzen mit den Aufwand abgewogen werden, um entscheiden zu können, ob eine Anforderung umgesetzt wird und ob der Prototyp um diese erweitert werden soll. Der primäre Nutzen ist dabei immer noch die Unterstützung dieser Masterarbeit durch die Nutzungsszenarien der Datenanalyse und -visualisierung.
Aus Zeitgründen kann nicht jede Anforderung umgesetzt werden. Es wurde lediglich die Konfiguration der Anzeigen von Informationen über die Url implementiert, da diese Anforderung einen direkten Nutzen für die Verfasserin dieser Arbeit bietet. Ebenfalls wurden kleine Probleme bezüglich Benutzerfreundlichkeit gelöst, die keinen großen Aufwand in der Implementierung darstellen, um den Prototypen abzurunden und Verwirrungen für Erstnutzerinnen und -nutzer zu vermeiden.

5.3.3 Ergebnisse der semi-strukturierten Interviews

Während der Usability Tests sollte durch semi-strukturierten Interviews ebenfalls herausgefunden werden, ob eine Taxonomie und somit der Prototyp TSEK-Analyzer relevant für Software Ingenieurinnen und Ingenieure sein kann. Außerdem soll die Nutzergruppe besser kennen gelernt werden und erschlossen werden, inwiefern diese eine potentielle Nutzergruppe darstellt. Dabei soll das Nutzungsszenario zur Verortung der Forschung und Weiterbildung, welches bereits in der Sektion 5.2.3 dieses Kapitels angeschnitten wurde, evaluiert werden.

Es wurden Fragen gestellt, inwieweit und in welchen Formaten die Testperson sich über aktuelle akademische Beiträge informiert und ob eine Taxonomie ein hilfreiches Format wäre. Um Einblicke in den Beziehungen zwischen den in Grafik 5.2 abgebildeten Segmenten zu erlangen, wurde außerdem nach dem eigenen Verhältnis zur akademischen Welt gefragt und nach der eigenen Sicht, wie die Industrie mit der akademischen Welt interagiert, insbesondere in Bezug auf Wissenstransfer.
Da der Test zeitlich begrenzt war und da nicht alle Teilnehmenden tiefes Domänenwissen im Bereich Software Engineering haben, wurden nicht allen Teilnehmenden alle Fragen gestellt. Nichts desto trotz, konnten trotz der kleinen Menge an Antworten

verschiedene Perspektiven erschlossen werden und Phänomene erkannt werden.

Aus den verschiedenen Beträgen der semi-strukturierten Interviews lässt sich die Gemeinsamkeit erkennen, dass die Befragten nicht regelmäßig akademische Arbeiten lesen, sondern andere Formate und Medien über die aktuellen Beiträge in Software Engineering bevorzugen, wie Podcasts, Artikel, Zeitschriften und Bücher oder Blogposts und Empfehlungen auf GitHub oder sozialen Medien, wie Twitter.
Ein Befragter besucht gerne Meetups und technische Konferenzen, die jedoch nicht unbedingt einen Bezug zu akademischen Arbeiten haben. Ein anderer Befragter hat zwar noch keine Konferenz besucht, jedoch hat er beobachtet, dass vor Allem Konferenzen einen großen Beitrag leisten, um Wissen in Unternehmen zu vermitteln.

Die Teilnahme an Konferenzen kann vom Arbeitgeber gefördert werden. Oft sind Unternehmen auch Sponsoren von Konferenzen. Jedoch ist unklar, inwiefern diese Konferenzen einen akademischen Hintergrund haben oder inwiefern ein Austausch zwischen den verschiedenen Segmenten, die in der Grafik 5.2 abgebildet sind, besteht. Ebenfalls ist unklar, inwieweit die von den Befragten aufgelisteten Formate und Medien einen akademischen Hintergrund haben. In Kontrast zu akademischen Arbeiten, ist in diesen Formaten kein Standard gegeben, Quellen transparent aufzuzeigen, sodass der akademische Hintergrund nicht immer erkennbar ist.

Als zweite Gemeinsamkeit aus den verschiedenen Beiträgen der semi-strukturierten Interviews macht sich bemerkbar, dass kaum Wissenstransfer mit der akademischen Welt besteht. Ein Befragter bedauert dies, da er sehr mit der akademischen Welt sympathisiert. Er kann schlecht einschätzen, wie nah die akademische Welt an der Praxis ist und meint das vieles auch in der Industrie geboren wird. Generell sprechen ihn persönlich auch mehr praxisbezogene Beispiele an, da seiner Meinung nach die Softwareentwicklung eine Disziplin ist, die sich immer an der Praxis messen lassen muss.
Ein anderer Befragter steht zwar noch in Kontakt zu Professoren seiner ehemaligen Hochschule, jedoch besteht kein fachlicher Austausch. Allerdings sieht er eine hohe Korrelation zwischen Forschung und Praxis.

Vermutung: Es können verschiedene Faktoren eine Rollen spielen für den wenigen Austausch zwischen den Segmenten Unternehmen und Universitäten, wie beispielsweise Größe des Unternehmens, Förderungsmöglichkeiten im Unternehmen, bürokratische Vorgänge und Strukturen oder Studienprogramme der Universitäten. Um diese Faktoren aufzudecken, müsste der Bedarf sowie das Interesse von Unternehmen und Universitäten zum Wissenstransfer und gegenseitigen Austausch untersucht werden.

Eine Befragte hat Möglichkeiten erläutert, inwiefern ein Arbeitgeber seine Mitarbeiterinnen und Mitarbeitern fördern kann um auf den aktuellen Stand des Bereichs

Software Engineering zu bleiben und um den Kontakt mit der akademischen Welt zu ermöglichen. Zum Einen durch die Teilnahme oder Leitung von Schulungen und das Verfassen von Blogeinträgen über ein aktuelles Thema, aber auch durch das Halten von Vorträgen an Universitäten.
Bei den ersten zwei Möglichkeiten ist, wie bei den Konferenzen der akademische Hintergrund unklar. Schulungen und Blogeinträge können aus praktischen Erfahrungen aus dem Unternehmen heraus entstehen und nicht immer muss eine tiefe Recherche erfolgen oder akademische Arbeiten vorgestellt werden.
Bei der dritten Möglichkeit wird sich aktiv zwischen den Segmenten Universität und Unternehmen ausgetauscht. Dabei sind meist die Untersegmente Studierende, Software Engineers als Gastvortragende und eine Professorin oder ein Professor, als Gastgebende, involviert. Ein anderer Befragter hat festgestellt, dass es gängig ist, dass Studierende ihre Abschlussarbeiten in Unternehmen schreiben, und somit Wissenstransfer in Unternehmen fördern.

Vermutung: Der aktive Austausch zwischen den zwei Segmenten Universität und Unternehmen durch Vorträge oder die gemeinsame Betreuung von Abschlussarbeiten hängt vermutlich viel von der Bereitschaft einzelner Personen ab und von persönlichen Beziehung, vor allem wenn noch keine standardisierten Abläufe oder Kommunikationswege vorhanden sind. Ob eine Taxonomie in der Kommunikation zwischen den Segmenten unterstützen kann, kann nur gemutmaßt werden. Eher kann eine Taxonomie bei der Verortung der akademischen Arbeiten helfen.

Ein Befragter hätte Interesse an einer Taxonomie, da sich sein Verständnis der Softwareentwicklung häufig in einem Graph darstellt und grafische Darstellungen von Strukturen und Beziehungen ihn helfen sich schneller zu orientieren.
Die Taxonomie könnte ein Gegenstück zu einen Technologieradar[21] sein um akademische Arbeiten zugänglicher zu machen und den Stand der Forschung leichter zu verorten. Abhängig von den Kontexten, für welche sich Software Ingenieurinnen und Ingenieure interessieren können sie aktuelle akademische Arbeiten finden. Dadurch können sie sich weiterbilden, recherchieren und mehr Expertise in ihren Interessenfeld gewinnen. Außerdem könnte das Interesse an Konferenzen mit akademischen Hintergrund, wie die ICSE 2020 und an akademische Arbeiten erhöht werden.

Schlussfolgerung: Es kann geschlussfolgert werden, dass die Taxonomie zur Weiterbildung und Verortung der Forschung für Praktizierende von Nutzen sein kann. Akademische Arbeiten in andere Formate zu übersetzen oder über geeignete Medien zu verbreiten kann das Interesse von Praktizierenden wecken und den Wissenstransfer zwischen den akademischen und industriellen Welt fördern.

[21] Ein Technologie Radar ist eine Visualisierung von Technologien, welche unter verschiedenen Gesichtspunkten eingeschätzt werden.

5.4 Evaluation der Taxonomie

Die identifizierten Kontexten aus der Literaturstudie und die resultierte Taxonomie soll durch die Befragung der Autorinnen und Autoren der untersuchten Arbeiten evaluiert werden. Diese wurden mittels einer Umfrage mit zehn Fragen gebeten, die Kontexte ihrer Arbeiten und die Taxonomie zu evaluieren. Zusätzlich wurde nach der eigenen Meinung zur expliziten Kontextbeschreibung gefragt und zu der Vorgehensweise und Herausforderungen bei der Evaluation der Verallgemeinerbarkeit und Relevanz in der eigenen Forschung. Am Schluß wurden Fragen zur Datenerhebung gestellt.

Die Umfrage, sowie das Anschreiben an die Autorinnen und Autoren, ist im Anhang dieser Arbeit in der Sektion 7.5 zu finden. Die Umfrage wurde durch ein Google Formular umgesetzt und die Autorinnen und Autoren hatten zehn Tage Zeit an diese teilzunehmen.

Insgesamt wurden 240 Autorinnen und Autoren der 59 gesichteten Arbeiten angeschrieben. Allerdings spiegelt dies nicht die Gesamtheit der Autorinnen und Autoren wider. In 8 Arbeiten sind keine Mailadressen angegeben und in drei Arbeiten lediglich von den Hauptautorinnen und -autoren. Im ersten Fall wurden die Mailadressen der Hauptautorinnen und -autoren ermittelt und lediglich diese wurden für die 8 Arbeiten kontaktiert. 15 Autorinnen und Autoren waren unter ihrer angegebenen Mailadresse nicht erreichbar. Auf Vollständigkeit wurde hierbei verzichtet und deren Mailadressen wurden nicht ermittelt.

5.4.1 Ergebnisse der Umfrage

Letztendlich haben von den 225 erreichten Autorinnen und Autoren 29 von 26 Arbeiten an der Umfrage teilgenommen und ihre Meinung geteilt.

Auf die erste Frage der Umfrage zur Meinung der Autorinnen und Autoren über die identifizierten Kontexte ihrer Arbeit, haben 12 Teilnehmende angegeben der Kontext ihrer Arbeit ist gut reflektiert. Ein Mal wurde der identifizierte Kontext einer Arbeit nicht zugestimmt und 7 Mal wurde teilweise nicht zugestimmt. Fünf Mal wurde angegeben, dass der Kontext der Arbeit weiter sei und vier Mal, dass der identifizierter Kontext zu weit sei.

Auf die zweite Frage, wie die Teilnehmenden den Kontext ihrer Arbeiten charakterisieren, haben 20 Teilnehmende geantwortet, sodass teilweise ihre Antwort auf die erste Frage ersichtlicher wurde. Von diesen Kommentaren wurden acht Verbesserungsvorschläge angewandt. 8 paper spezifische Kontexte wurden hinzugefügt. Außerdem wurden die Zwecke von Kontexten innerhalb zwei Arbeiten neu bestimmt. Auf die Taxonomie

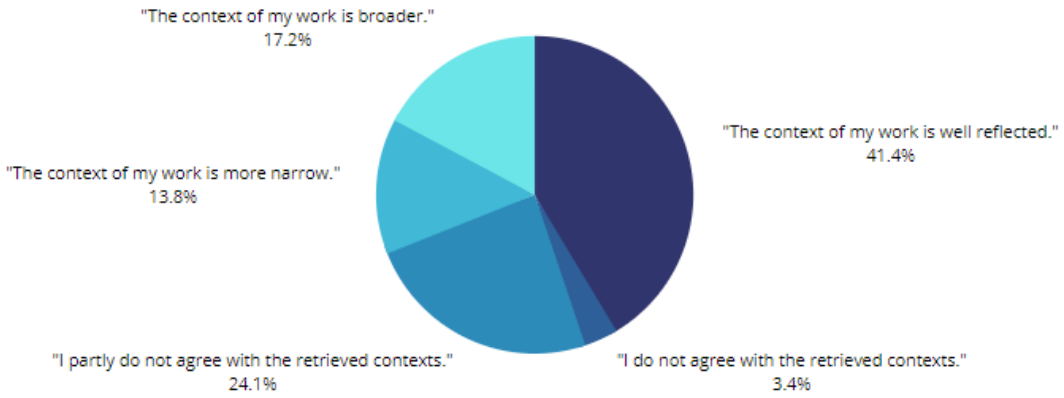


Figure 5.2: Antwort auf die erste Frage der Umfrage: *'After analysing your work's data, which of the following sentences reflects your opinion about the retrieved contexts the most?'*

hatte diese Änderungen lediglich Auswirkungen an einer Stelle, da einige vorgeschlagene Kontexte bereits in der Taxonomie vorhanden waren.

Die dritte Frage der Umfrage bezog sich auf die gesamte Taxonomie. 20 Teilnehmende haben diese als eher positiv empfunden. Vermehrt kamen in den Antworten Wörter wie angemessen (*"reasonable"*), treffend (*"accurate"*) und akzeptabel vor. Unter anderen haben zwei Teilnehmende den Detaillierungsgrad der Taxonomie gelobt.
Im Gegensatz dazu wurden fünf eher negative Antworten eingereicht, dass die Taxonomie schwer zu interpretieren sei, verwirrend, ambivalent oder begrenzt (*"limiting"*) sei. Eine Antwort hat vorgeschlagen, die Arbeiten anderer Konferenzen zu analysieren, sodass die Taxonomie umfassender wird. In einer anderen Antwort wurde auch darauf hingewiesen, dass es drei große Unterbäume bezüglich *Testing*, *Bug Localization* und *Bug Handling* gibt, jedoch wenig Kontext zu Themen wie *Language* oder *Pattern Design*. Das liegt daran, dass die gesichteten Arbeiten eher in diesen Kontexten liegen und dass die Taxonomie lediglich die gesichteten Arbeiten reflektiert und auch nur Arbeiten, die an der ICSE 2020 angenommen wurden.

Innerhalb der Antworten auf der dritten Frage wurden neun Verbesserungsvorschläge bezüglich der Taxonomie geäußert. Davon wurden sieben umgesetzt. Außerdem wurde die Taxonomie erneut evaluiert und an zwei Stellen umstrukturiert.

Schlussfolgerung : Durch die Änderungen an der Taxonomie aus dem Feedback der Autorinnen und Autoren, kann geschlussfolgert werden, dass die Autorinnen und Autoren eher in der Anfertigung der Taxonomie einbezogen werden sollten, anstatt in der Evaluierung von dieser.

Auf die vierte Frage, ob die Teilnehmenden die Kontextbeschreibung als wertvoll empfinden, haben 17 Teilnehmende positiv reagiert und viele Teilnehmende haben angemerkt, inwiefern die Taxonomie nützlich sein könnte. Im Gegensatz dazu wurde in einer Antwort die Taxonomie nicht als praktisch wahrgenommen, um eine neue, spezifische Arbeit zu schreiben.
Unter den anderen Antworten wurde es jedoch als nützlich erachtet, zum einen um relevante Arbeiten zu finden, insbesondere neu, veröffentlichte Arbeiten, die in dem eigenen Feld liegen. Es wurde erklärt, dass zu viele Arbeiten veröffentlicht werden und eine Taxonomie bei der Entscheidung helfen könnte, welche Arbeiten zu lesen. Die Taxonomie kann eine Übersicht des Software Engineering Feldes sein und einen gemeinsamen Wortschatz schaffen, um Ideen leichter auszutauschen.

Zwei Teilnehmende haben auf die Schlüsselwörter bzw. die CCS Konzepte [22] hingewiesen, die Autorinnen und Autoren angeben, und sind sich unsicher, inwieweit eine Taxonomie Mehrwert liefert. Dies ist bereits in der Sektion 7.4 in einem Interview mit den Gutachterinnen und Gutachter der ICSE aufgekommen. Am Ende dieser Sektion wird darauf näher eingegangen.

Außerdem wurde angemerkt, dass es schwierig sei, mit einer vordefinierten Taxonomie den eigenen Kontext einer Arbeit festzulegen. Diese Aussage stützt die obige Schlussfolgerung. dass die Autorinnen und Autoren bei der Anfertigung der Taxonomie integriert werden. Mit einer agilen Taxonomie haben Autorinnen und Autoren die Möglichkeit die Taxonomie zu erweitern und umzustrukturieren, sodass der Kontext ihrer Arbeit reflektiert wird.

Auf die fünfte Frage bezüglich der Evaluation der Verallgemeinerbarkeit von Ergebnissen, sind viele verschiedene Antworten eingegangen.
Vermehrt wurde darauf hingewiesen, dass die Verallgemeinerbarkeit abhängig von der Repräsentativität der Daten sei und somit von einer breiten Auswahl der evaluierten Subjekte von Belang ist. Die Teilnahme von industriellen Partner bzw. die Durchführung in einer realistischen Umgebung sei hilfreich, da Open Source Daten begrenzt seien und nicht generalisierbar auf andere Datensätze. Bei der Auswahl können etablierte Benchmarks unterstützen, wenn diese jedoch nicht verfügbar sind, wurde empfohlen reale Subjekte zur Evaluation zu nutzen und verschiedene Anwendungsfelder zu beachten. Es wurde beschrieben, dass darauf geachtet wird, dass die Ergebnisse nicht zu angepasst auf die durchgeführte Studie seien und dass evaluiert wird, inwieweit sich die Ergebnisse auf andere Subjekte anwenden lassen.

Eine Antwort hat darauf hingewiesen, dass neu vorgeschlagene Algorithmen oder Methoden mittels wenig Engineering Aufwand einfach auf andere Domänen anwendbar seien.

[22] https://dl.acm.org/ccs

Zwei Teilnehmende stützen sich auf die verwandte Literatur und nutzen Best Practises oder Kriterien von vorherrschenden Arbeiten, um die Verallgemeinerbarkeit ihrer Arbeit zu evaluieren. Eine weitere Antwort beschreibt, dass die Einschränkungen und die Skalierbarkeit der Ergebnisse abgeschätzt werden, sowie die Anwendbarkeit der gemachten Annahmen bei theoretischen Ergebnissen.
In eine Antwort wurde auf einen Leitfaden zur externen Gültigkeit von Wohlin [23] hingewiesen. In einer anderen Antworten wurde zugegeben, dass ein Standard um die Verallgemeinerbarkeit von Ergebnissen empirisch zu prüfen nicht bekannt sei und in einer weiteren Antwort wurde das Fehlen von klaren und expliziten Standards als Herausforderung angesehen.

Auch auf die letzte Frage bezüglich der Evaluation der Relevanz von Ergebnissen, sind viele verschiedene Antworten eingegangen.

Es wurde auf die Meinung und Zustimmung von Praktizierenden hingewiesen. Es wurden die Implikationen für Praktizierende beachtet und die Ergebnisse von der Perspektive der Praktizierenden evaluiert. Dies kann durch verschiedene Methoden erreicht werden, wie Experten Evaluation, Nutzerstudien, Interviews und Workshops. Dabei gebe es jedoch verschiedene Herausforderungen. Zum einen genug Antworten zu bekommen, zum anderen können die Kosten hoch sein und es sei schwierig Nutzerstudien korrekt durchzuführen. Außerdem können sich die Meinungen von Praktizierenden unterscheiden.
Einige Antworten referieren wieder auf die Auswahl realer Subjekte, sowie auf das Vergleichen mit bereits bestehender Literatur. Des Weiteren werden die Anwendung von Metriken, die Benutzerfreundlichkeit der Techniken, die Anwendbarkeit in der Umgebung der Endnutzerinnen und -nutzer, sowie die Motivation der Arbeit als Faktoren für die Relevanz erwähnt. In einer Antwort wird die Akzeptanz der Arbeit auf der ICSE als Maß genommen, dass die Arbeit relevant für die SE-Community sei.

Schlussfolgerung : Aus diesem Potpourri von Antworten bezüglich der Verallgemeinerbarkeit und Relevanz lässt sich schließen, dass keine Standards zur Evaluierung dieser Kriterien bestehen und Autorinnen und Autoren verschiedene Ansätze verfolgen. Diese Schlussfolgerung ist bereits aus einem der Interviews mit den Gutachterinnen und Gutachtern deutlich geworden.

[23] https://scholar.google.de/citations?user=gGLdjasAAAAJhl=deoi=sra

5.4.2 Vergleich der Taxonomie

In dieser Sektion wird die Taxonomie mit bereits bestehenden Taxonomien und Ergebnisse verwandter Arbeiten verglichen.

Das *Computing Classification System*(CSS)[24] ist ein Klassifikationssystem für die Informatik. Es wurde von der *Association for Computing Machinery* (ACM), die erste wissenschaftlichen Gesellschaft für Informatik, erstellt. Dadurch dass Autorinnen und Autoren ihre Arbeits mittels den CCS Konzepten klassifizieren können Leserinnen und Leser diese Arbeit und verwandte Arbeiten leichter finden. Denn das CCS System ergänzt die bibliografische Suche von ACM. [1] [60]

Bei dem Vergleich der Taxonomie und den CCS Konzepten fällt auf, dass die CCS Konzepte ein breiteres Feld abdecken. Dies liegt daran, dass die Taxonomie lediglich die 59 gesichtete Arbeiten der ICSE 2020 umfasst.
Die Taxonomie jedoch taucht tiefer in einen Kontext ein. Die CCS Konzepte haben beispielsweise keine Unterkonzepte für Testing. Auch werden Werkzeuge, Frameworks und Programmiersprachen nicht explizit genannt.

In der Arbeit *"Context in Industrial Software Engineering Research"* [44] haben Kai Petersen und Claes Wohlin sechs Kategorien als Checkliste zur Kontextbeschreibung extrahiert, *product*, *processes*, *practices, tools and techniques*, *people*, *organization*, und *market*. Die ersten vier Kategorien und teilweise auch deren Unterkategorien können in der Taxonomie wiedergefunden werden. Wie bei den CCS Konzepten ist die Taxonomie konkreter und stellt mehr Beispiele zur Verfügung.

Im Gegensatz wird in der Arbeit *"A Taxonomy of Software Types to Facilitate Search and Evidence-Based Software Engineering"* [13] von Andrew Forward und Timothy C. Lethbridge sehr konkret auf die Softwaretypen, insbesondere deren Anwendungsdomänen eingegangen. Diese Taxonomie kann mit den Dimensionen Anwendungen und Werkzeuge verglichen werden, wobei sie tiefer in die Kontext eingeht und umfassender ist. Dabei fällt auf, dass die Taxonomie dieser Masterarbeit unvollständig ist und nur die Kontexte von den gesichteten Arbeiten widerspiegelt.

[24] https://dl.acm.org/ccs

6 Abschluss

Im Folgendem werden die Ergebnisse aus den vorherigen zwei Kapitel zusammengefasst. Abschließend werden diese, sowie die Relevanz und die Verallgemeinerbarkeit dieser Masterarbeit, diskutiert. Außerdem wird ein Ausblick auf den weiteren Forschungsbedarf gegeben.

6.1 Zusammenfassung der Ergebnisse

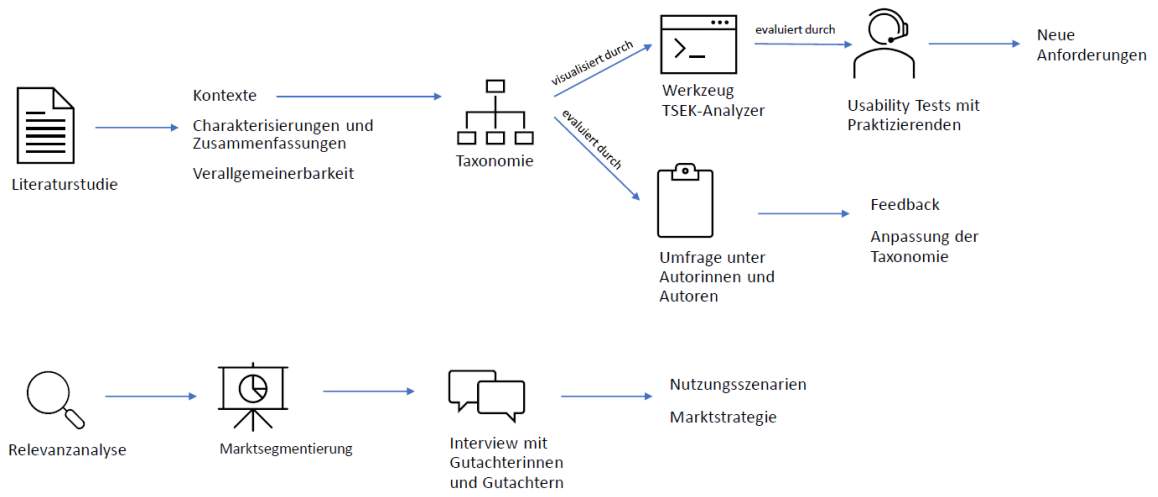


Figure 6.1: Aktivitäten und Ergebnisse innerhalb dieser Masterarbeit

Mittels einer Literaturstudie von 59 technischen Arbeiten der ICSE wurde eine Taxonomie angefertigt, die aus sechs Bäumen besteht. Um diese zu visualisieren wurde das Werkzeug *TSEK-Analyzer* gebaut, welches innerhalb Usability Tests von Software Engineers und Requirements Expertinnen und Experten evaluiert wurde. Dabei wurden vier Anforderungen an das Werkzeug aufgedeckt.

Mittels einer Umfrage an die Autorinnen und Autoren der gesichteten Arbeiten wurde der Kontext von acht Arbeiten korrigiert und die Taxonomie an sieben Stellen angepasst. Durch eine Segmentierung des Marktes und semi-strukturierten Interviews mit Gutachterinnen und Gutachtern, wurde die Relevanz einer Taxonomie evaluiert. Dabei wurden acht Nutzungsszenarien für die Taxonomie herausgearbeitet und in einer Marktstrategie, die in Sektion 6.4 diese Kapitels erläutert wird, priorisiert.
Zunächst werden jedoch in der folgenden Sektion die Schlussfolgerungen, die innerhalb der vorherigen zwei Kapitel gezogen wurden, in Beziehung miteinander gesetzt.

6.2 Beantwortung der Forschungsfragen

Innerhalb dieser Sektion werden die Forschungsfragen, die in Kapitel 3 zum Vorgehen aufgestellt wurden, beantwortet. Dazu wurden die Ergebnisse und Schlussfolgerungen, die im Kapitel 4 zur Durchführung gezogen wurden, betrachtet.

- In wie weit beschreiben Autorinnen und Autoren den Kontext ihrer Arbeit?
- Sind andere Zwecke, als die in Tabelle 3.1 genannten, erkennbar?

Autorinnen und Autoren beschreiben ihren Kontext hinsichtlich verschiedener Zwecke. Während der Literaturstudie wurden Kontexte zum Thema, Details, Ein-/Abgrenzung und zur Verallgemeinerbarkeit identifiziert. Dadrüber hinaus wurden zwei weitere Zwecke, als die in Tabelle 3.1 genannten, Erweiterung und Anwendung eingeführt und weitere Kontexte wurden identifiziert.
Es ist aufgefallen, dass die meisten Kontexte zu den Zwecken Details (28%), Anwendung (24%) und Abgrenzung (20%) zugeordnet wurden und die meisten Kontexte dabei in den folgenden Sektionen vorkommen: Einleitung (24%), Abstract (17%) und Related Work (16%).

- In wie weit diskutieren Autorinnen und Autoren über die Verallgemeinerbarkeit ihrer Ergebnisse?
- Bestehen Lücken in der Diskussion zur Verallgemeinerbarkeit?
- Wie unterscheiden sich die Kontexte zum Zweck der Verallgemeinerbarkeit zu den Kontexten anderer Zwecke?

In der Analyse bezüglich der Verallgemeinerbarkeit, welche in Sektion 4.2 beschrieben wurde, wurde geschlussfolgert, dass die externe Gültigkeit und somit auch die Verallgemeinerbarkeit und der größere Kontext einer Arbeit uneinheitlich von Autorinnen und Autoren beschrieben wird. Zum Teil wurde diese gar nicht diskutiert. Die Hälfte der Arbeiten verlässt sich auf die Repräsentativität der untersuchten Subjekte, anstatt explizit und konkret den größeren Kontext einer Arbeit zu beschreiben, über die Verallgemeinerbarkeit zu diskutieren und Beweise aufzuführen.
Die gefundenen Kontexte mit dem Zweck Verallgemeinerbarkeit unterscheiden sich zu den identifizierten Kontexten anderer Zwecke in der Hinsicht, dass diese in vagen Aussagen gefunden wurden und diese Aussagen nicht immer begründet oder erläutert wurden.
Es konnte geschlussfolgert werden, dass Lücken in der Diskussion zur Verallgemeinerbarkeit bestehen.

Innerhalb des Interviews mit Gutachterinnen und Gutachter der ICSE 2020 und der Evaluation des Nutzungsszenario zum Verfassen und Einreichen einer Arbeit, ist erkennbar, dass Schwierigkeiten beim Beweisen der Verallgemeinerbarkeit vorliegen. Eine Methodik fehlt, die Autorinnen und Autoren unterstützt konkretere Aussagen zu

tätigen und diese zu beweisen, sodass eine explizitere Kontextbeschreibung erschwert wird.
Innerhalb der Umfrage mit Autorinnen und Autoren der gesichteten Arbeiten ist erkennbar, dass verschiedene Ansätze verfolgt werden, um die Verallgemeinerbarkeit, so wie die Relevanz einer Arbeit, zu beschreiben.
Eine Taxonomie kann bei der Verortung der Arbeiten innerhalb eines Kontextes helfen, jedoch nicht bei der Beweisführung. Es schafft eine Terminologie, die beim Einschätzen des Kontextes helfen kann und somit auch bei der Kontextbeschreibung, aber es schafft keine Methodik den Kontext einer Arbeit zu beweisen. Daher ist weitere Forschung nötig, um eine solche Methodik zu entwickeln.
Ohne eine solche Methodik, kann eine Taxonomie nicht als Standard eingeführt werden und eine explizitere Kontextbeschreibung kann nicht als Qualitäts- oder gar Bewertungskriterium im Gutachtensprozess angesehen werden, ohne Arbeiten einzuschränken.
Somit ist eine solche Methodik Vorraussetzung für einer tieferen Taxonomie von Software Engineering Kontexten, die in begründeten und konkreten Aussagen gefunden werden.

6.3 Diskussion zur Relevanz und Verallgemeinerbarkeit

Da lediglich 59 technische Arbeiten von lediglich einer Software Engineering Konferenz von lediglich einem Jahr untersucht wurden, repräsentiert die Taxonomie nicht das komplette Feld des Software Engineering, sondern lediglich die Kontexte der gesichteten Arbeiten.
Die ICSE besteht schon seit 1975 und ist eine sehr etablierte Plattform für Software Engineering, sodass die Arbeiten, die innerhalb dieser Konferenz veröffentlicht werden, die aktuellen Beiträge und die aktuellen Trends in Software Engineering darstellen. Es kann angenommen werden, dass die Taxonomie ein Teil der Trends, die auf der ICSE 2020 verfolgt werden, darstellt.

Für spezialisierte Konferenzen oder Zeitschriften kann eine Teilbaum der Taxonomie interessant sein. Der Grad der Anwendbarkeit hängt davon ab, inwieweit sich die Kontexte der spezialisierten Konferenzen oder Zeitschriften überschneiden. Eventuell ist der Teilbaum nicht genug ausgearbeitet und zu oberflächlich. Doch durch die Erweiterung der Taxonomie durch die Sichtung weiterer Arbeiten, können diese Teilbäume spezialisiert werden.

Für umfassendere Konferenzen oder Zeitschriften als die ICSE, könnte die angefertigte Taxonomie nur für einen Teil der Arbeiten relevant sein. Auch in diesem Fall kann die Taxonomie durch die Sichtung weiterer Arbeiten, die nicht durch die Taxonomie abgedeckt werden, erweitert werden, sodass sie nicht nur an Tiefe, sondern auch an Vollständigkeit gewinnt.

Eine Erweiterung der Taxonomie ist unabdingbar, damit die Taxonomie an Relevanz und Verallgemeinerbarkeit gewinnt. Die hier angefertigte Taxonomie kann als Grundstein gesehen werden. Jedoch sollte diese nicht starr sein, sondern agil, sodass sie angepasst und umstrukturiert werden kann.

6.4 Nutzungsszenarien und Marktstrategie

Weitere Schritte zu einer vollständigeren Taxonomie von Software Engineering Kontexten werden im Folgenden innerhalb einer Marktstrategie erläutert. Diese Strategie stellt ein Vorgehen dar, um die Taxonomie nicht nur zu vervollständigen, sondern auch innerhalb den verschiedenen Segmenten zu verbreiten. Die acht Nutzungsszenarien, die im vorherigen Kapitel herausgearbeitet wurden, sowie die Segmente und Interessengruppen, in die der Markt eingeteilt wurde, steillen die Grundlager dieser Marktstrategie dar.

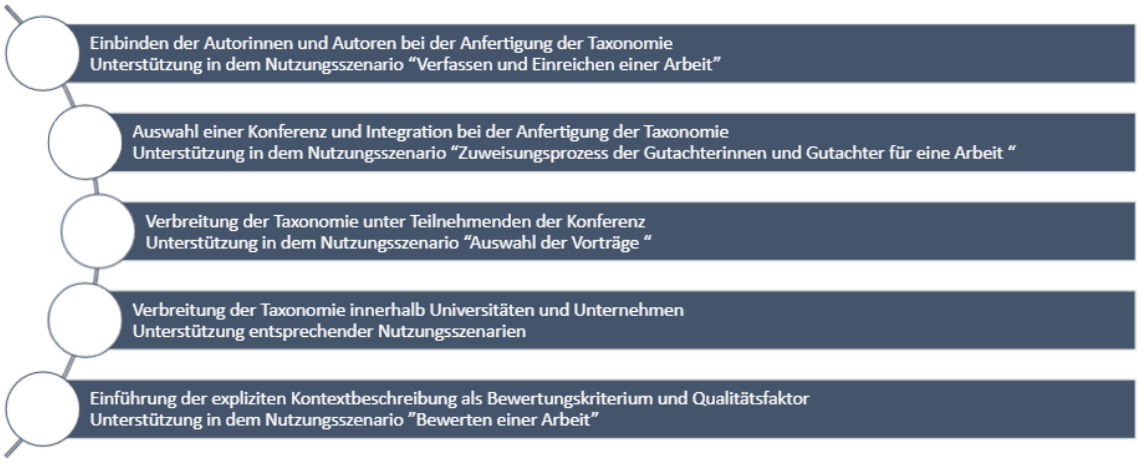
Die Zielgruppe der Marktstrategie sind die Autorinnen und Autoren, da dessen Nutzungsszenario das *Verfassen und Einreichen einer Arbeit* die explizitere Kontextbeschreibung beinhaltet, welche Voraussetzung für die Anfertigung einer tieferen Taxonomie ist, welche wiederum beim Verfassen von Arbeiten und bei der expliziteren Kontextbeschreibung von Nutzen sein kann.
Die Autorinnen und Autoren können in diesem Nutzungsszenario unterstützt werden, indem wie bereits erwähnt eine Methodik entwickelt wird, die beim Beweisen von der Verallgemeinerbarkeit einer Arbeit unterstützt und somit auch bei der Kontextbeschreibung.

Aus der Umfrage mit Autorinnen und Autoren wurde geschlussfolgert, dass diese früher in der Taxonomie integriert werden sollten. Durch das Einbinden von Autorinnen und Autoren in den Anfertigungsprozess, kann die Taxonomie an Qualität gewinnen. Anstatt einer Literaturstudie durchzuführen um eine Taxonomie anzufertigen, kann ein User-Centered Design Ansatz verfolgt werden, indem die Autorinnen und Autoren die Taxonomie aktiv mitgestalten.

Ein solcher Ansatz kann bei dem Einreichen von Arbeiten an einer Konferenz realisiert werden, indem Autorinnen und Autoren die Kontexte, in denen sich ihre Arbeiten bewegen, angeben müssen. Dazu könnte TSEK-Analyzer um entsprechende Funktionalitäten und Schreiboperationen erweitert werden, sodass Autorinnen und Autoren es nutzen können, um ihre Arbeit hinzuzufügen und ihre Kontexte in einer Taxonomie zu integrieren. Es ist denkbar, dass die Gutachterinnen und Gutachter die von den Autorinnen und Autoren bestimmten Kontexte evaluieren.
Um diesen Ansatz zu verfolgen, muss eine Konferenz gefunden werden, die bereit ist diesen Ansatz umzusetzen. Für die Konferenz könnte dies der Vorteil haben, dass durch eine Taxonomie der Zuweisungsprozess der Gutachterinnen und Gutachter für

eine Arbeit erleichtert wird, welches ein Nutzungsszenario darstellt.

Nachdem eine Taxonomie durch die Autorinnen und Autoren, sowie den Gutachterinnen und Gutachtern erstellt wurde, kann diese unter den Teilnehmenden der Konferenz genutzt werden, um das Nutzungsszenario Auswahl der Vorträge zu unterstützen. Dadurch wären alle drei Untersegmente des Segmentes Konferenz abgedeckt. Um die Nutzungsszenarien 4-7 zu unterstützen, sollte nach der Konferenz, diese Taxonomie innerhalb der anderen Segmente beworben werden.
Erst wenn sich die Taxonomie von Software Engineering Kontexten etabliert hat und vor Allem von Autorinnen und Autoren akzeptiert wird, kann das letzte Nutzungsszenario unterstützt werden und die explizite Kontextbeschreibung als Bewertungskriterium und Qualitätsfaktor hinzugenommen werden, bis es sich letztendlich zu einem Standard entwickeln kann. Es ist möglich, dass bis dahin sich die Taxonomie in den Folgejahren der Konferenz entwickelt und erweitert.



6.5 Ausblick

Als Fortsetzung dieser Abschlussarbeit können verschiedene Ansätze verfolgt werden.

Zum Einem ist es, wie bereits erwähnt, unabdingbar, dass eine Methodik entwickelt wird, um die Beweisführung bezüglich der Verallgemeinerbarkeit von Arbeiten zu unterstützen und somit die Kontextbeschreibung zu erleichtern.

Des Weiteren kann TSEK-Analyzer um mehr Funktionalitäten erweitert werden, um andere Nutzungsszenarien zu implementieren, sodass das Werkzeug für andere Personengruppen von Nutzen ist. Dabei ist die Verfolgung der in der Sektion 6.4 vorgestell-

ten Marktstrategie empfehlenswert, um einen User-Centered Ansatz zu verfolgen und um somit die potentiellen Nutzerinnen und Nutzer so früh wie möglich zu integrieren. Auch können die Anforderungen, die in dem Usability Tests aufgedeckt wurden, im TSEK-Analyzer umgesetzt werden.

Außerdem können die Annahmen und Vermutungen, die in dieser Masterarbeit aufgestellt werden, evaluiert werden. Am Interessantesten dabei ist die Beziehung zwischen den Segmenten Unternehmen und Industrie. Es kann geforscht werden, wie der Wissenstransfer gestärkt werden kann und ob eine Taxonomie dabei helfen könnte. Durch einen Vergleich von Software Engineering mit anderen Feldern der Informatik, wo die Kooperationen zwischen der Industrie und der Akademie stärker sind, können Missstände aufgedeckt werden und Einsichten erlangt werden.

Um die Ergebnisse dieser Arbeit zu evaluieren und die interne Gültigkeit dieser Arbeit zu stärken, kann eine Folgearbeit geleistet werden, die die Arbeiten der ICSE 2020 erneut sichtet und die Kontexte analysiert. Dabei kann die hier angefertigte Taxonomie um unentdeckte Kontexte erweitert werden und an Qualität gewinnen. Auch kann die Taxonomie durch die Sichtung weiterer Arbeiten, welche innerhalb anderer Konferenzen und Zeitschriften veröffentlicht wurden, erweitert werden. Durch die Identifizierung weiterer Kontexte in spezialisierteren Themengebieten wird die Taxonomie größer und das Feld Software Engineering wird vollständiger dargestellt. Jedoch kann dadurch die Taxonomie auch an Komplexität gewinnen, welches die Einordnung des Kontextes einer Arbeit erschweren könnte und den Nutzen der Taxonomie verringert.
Dahingegen kann analysiert werden, ob eine große Taxonomie für das Feld Software Engineering oder eine Taxonomie spezialisiert auf eine Konferenz von mehr Nutzen ist. Die in der Sektion 6.4 vorgestellte Marktstrategie bezieht sich auf den zweiten Fall, wobei es möglich ist eine innerhalb einer Konferenz angefertigte Taxonomie zu einer übergreifenden Taxonomie zu erweitern.

7 Anhang

7.1 Taxonomie

Im Folgendem wird die Taxonomie mittels Screenshots von dem Werkzeug *TSEK-Analyzer* repräsentiert. Wie bereits in der Sektion 4.4 erwähnt wurde, wurden sechs Dimensionen gebildet: Software Engineering Aktivitäten, Software Werkzeuge, Anwendungen, Daten, Human Aspects und arbeits-spezifische Kontexte. Die erste Dimension zu den Softwaren Engineering Aktivitäten wurde in vier Grafiken 7.1 bis 7.4 dargestellt und die Unterbäume zum Testing, Bug Handling und Fault Localization/Detection wurden ausgelagert.

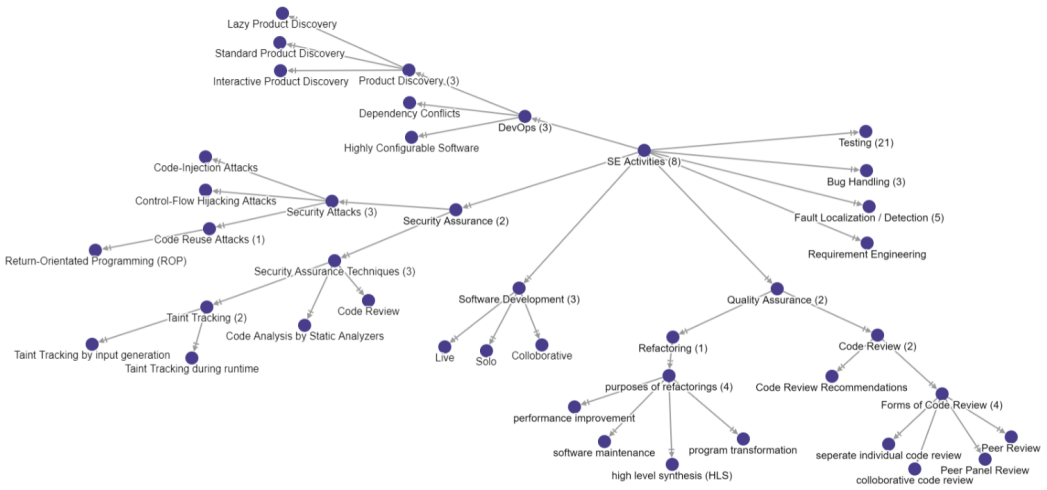


Figure 7.1: Taxonomie zu der Dimension: Software Aktivitäten

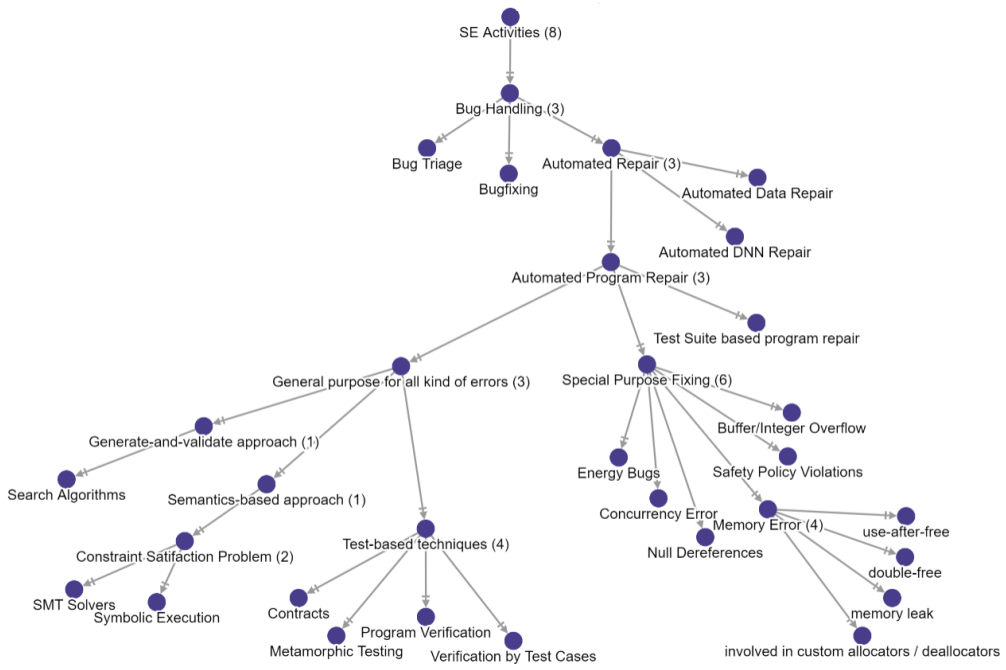


Figure 7.2: Taxonomie zu der Dimension: Software Aktivitäten > Bug Handling

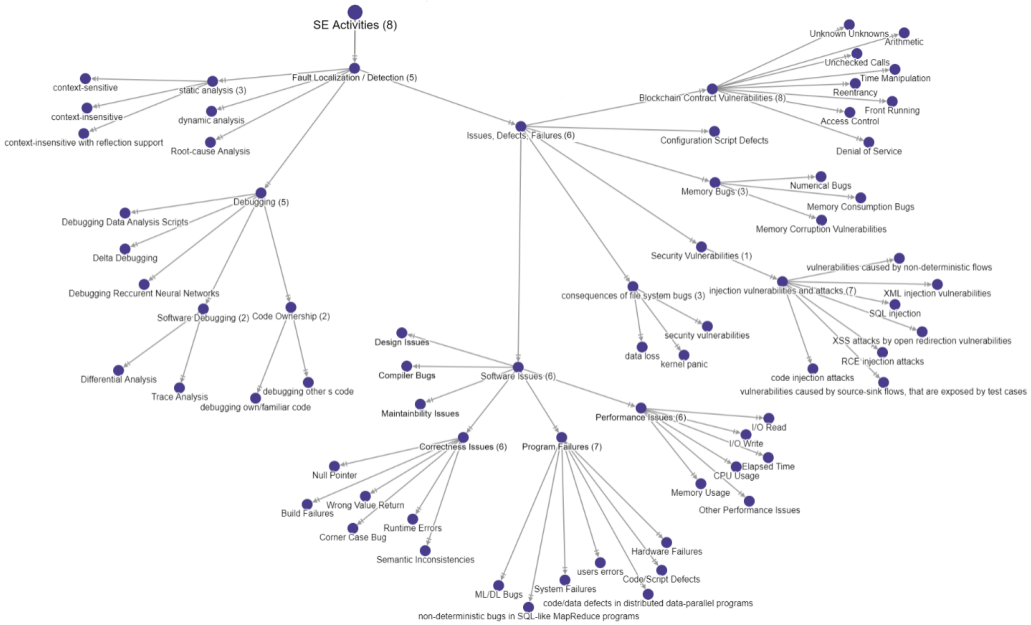


Figure 7.3: Taxonomie zu der Dimension: Software Aktivitäten > Fault Localization / Detection

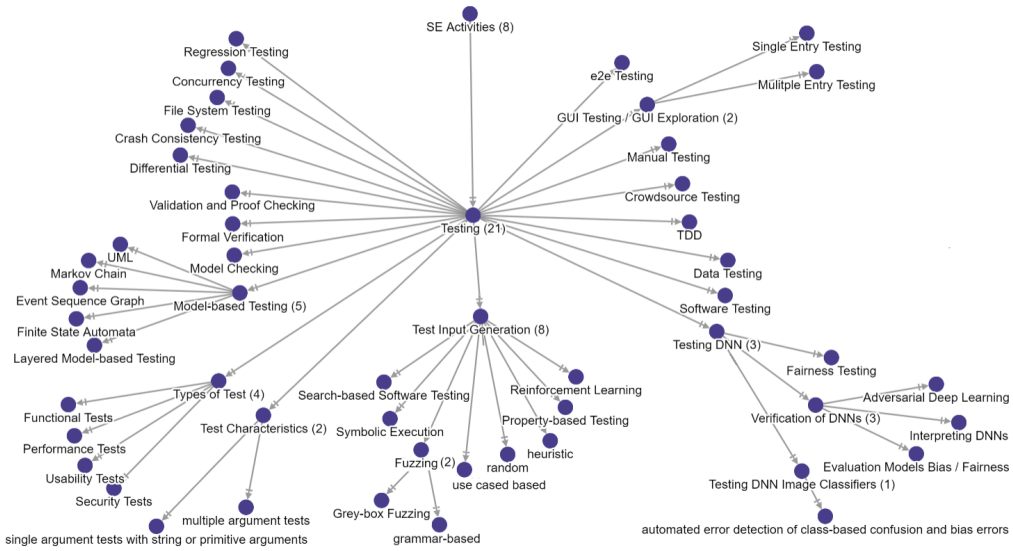


Figure 7.4: Taxonomie zu der Dimension: Software Aktivitäten > Testing

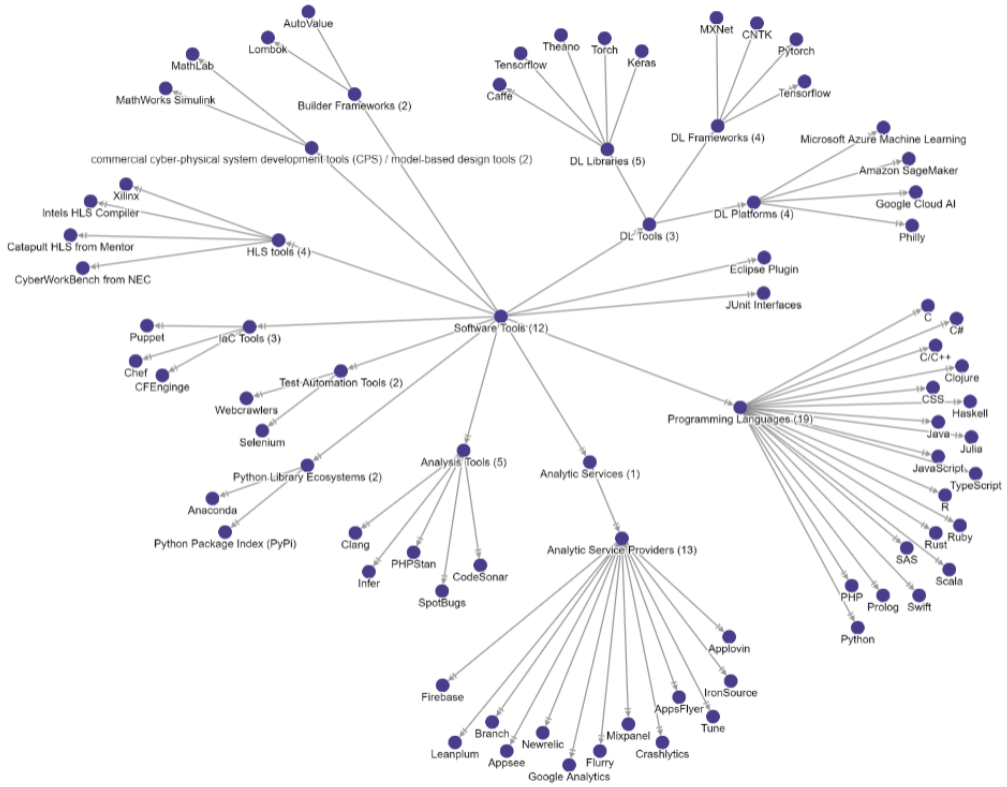


Figure 7.5: Taxonomie zu der Dimension: Software Werkzeuge

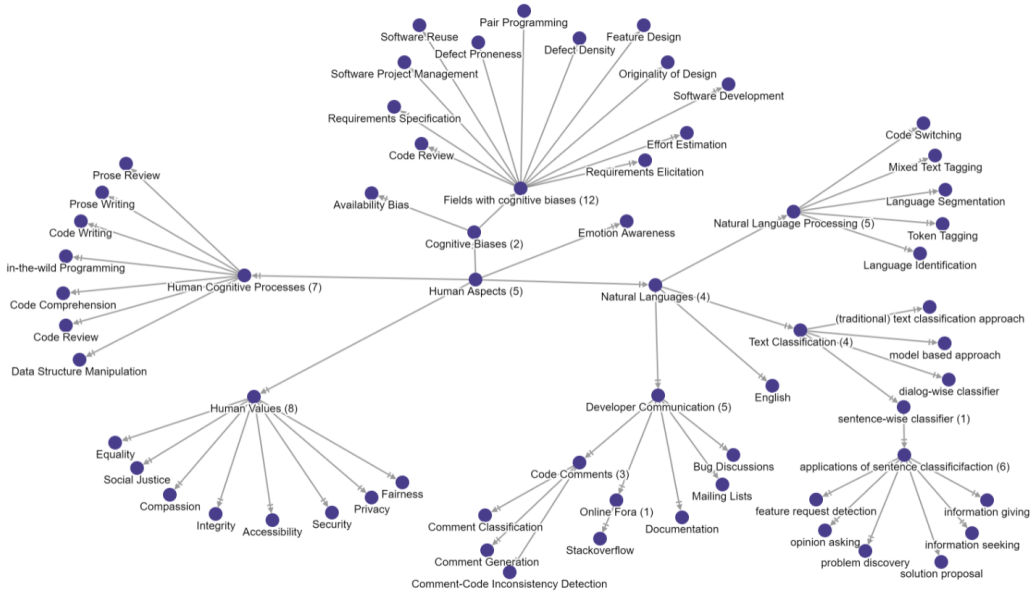


Figure 7.6: Taxonomie zu der Dimension: Human Aspects

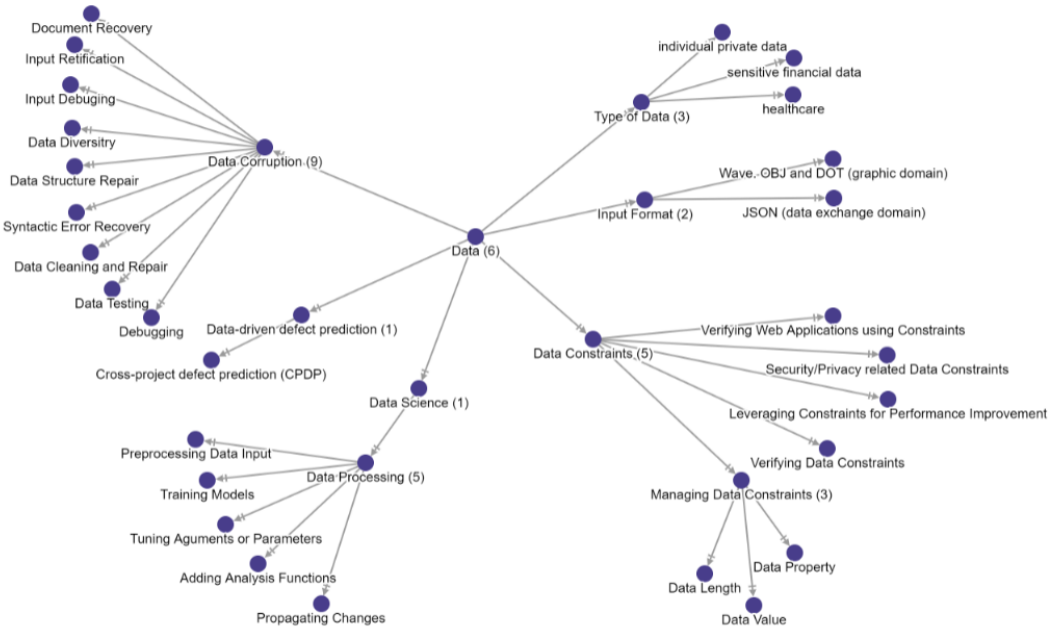


Figure 7.7: Taxonomie zu der Dimension: Daten

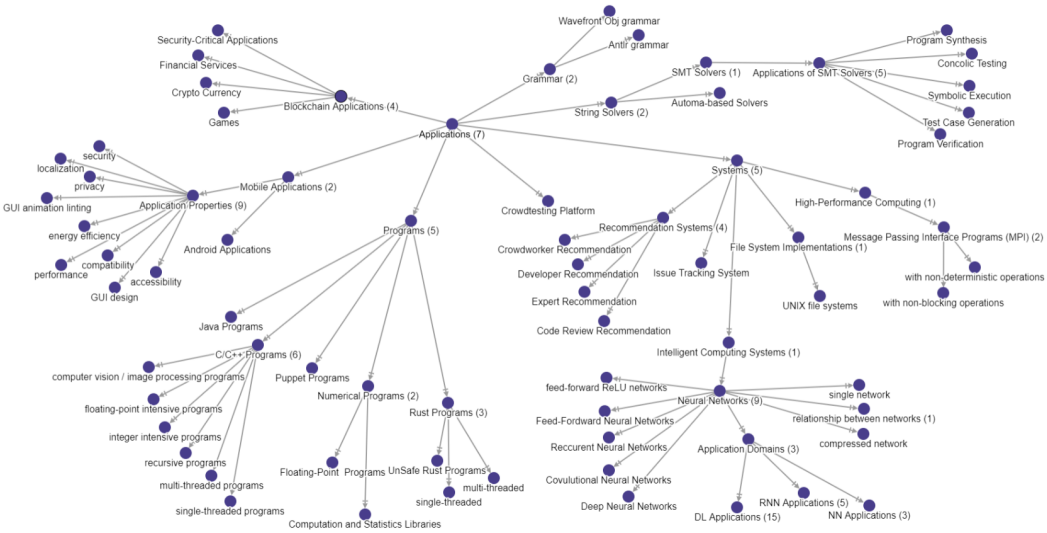


Figure 7.8: Taxonomie zu der Dimension: Anwendungen

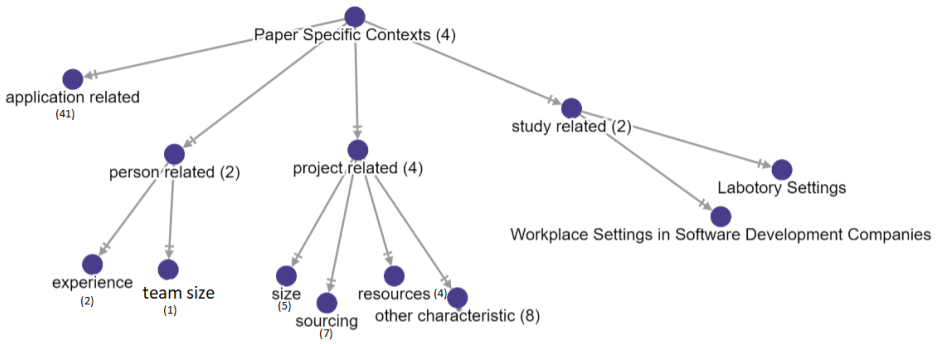


Figure 7.9: Taxonomie zu Paper-spezifischen Kontexten

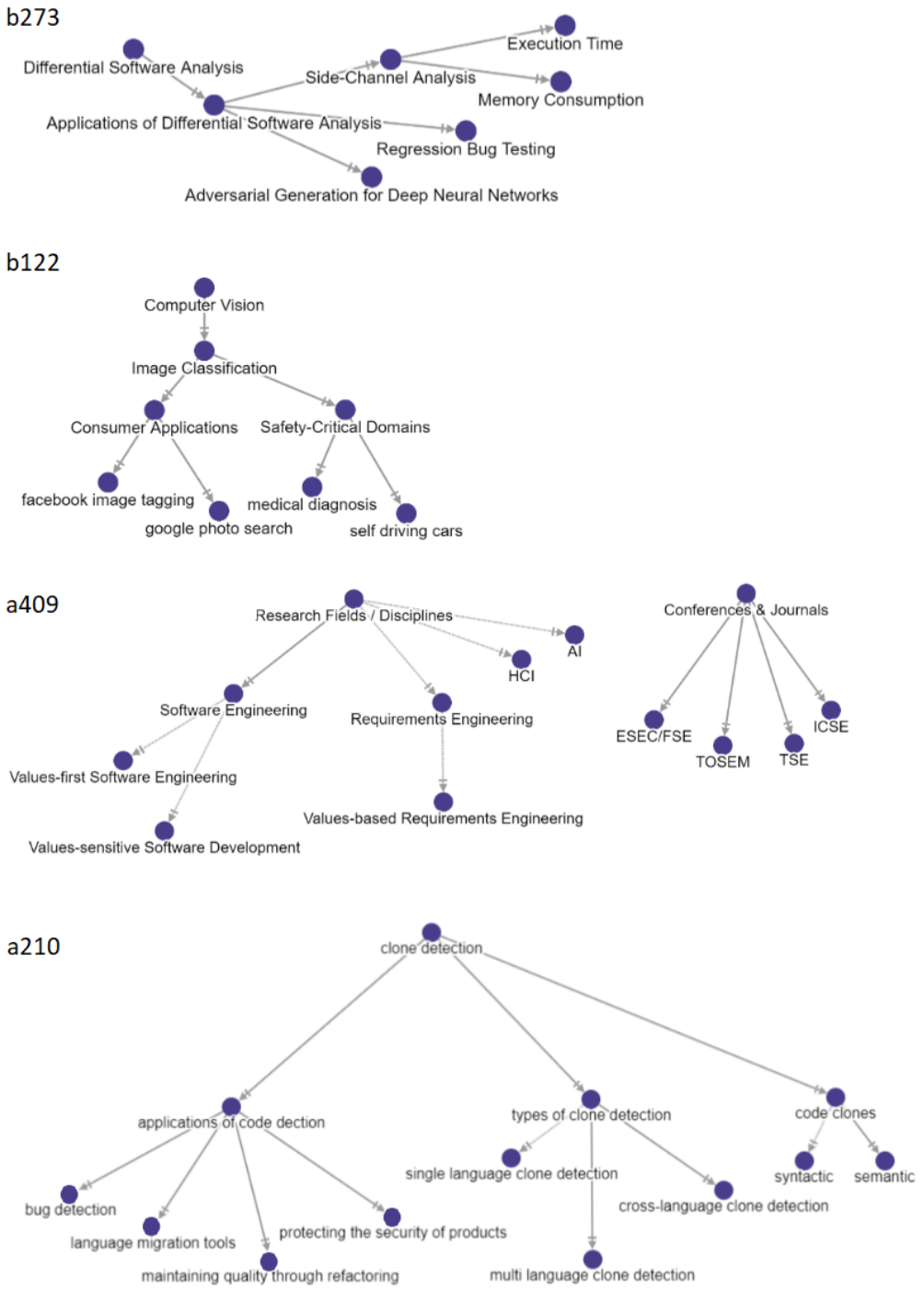


Figure 7.10: Kontexte, die nicht integriert in der Taxonomie sind

7.2 Übersicht über die gesichteten Arbeiten

Im Folgenden werden die gesichteten Arbeiten vorgestellt. Mittels Angabe der id einer Arbeit wird in dem entwickelten Werkzeug *TSEK-Analyzer* eine Einzelansicht dieser Arbeit und dessen Kontexte visualisiert. [25]

Context-aware In-process Crowdworker Recommendation id: b535 [57]
Die Autorinnen und Autoren dieser Arbeit haben einen Ansatz entwickelt, um Empfehlungen für *Crowdworker* zu optimieren, die die entwickelte Software testen. Dadurch können Kosten gespart werden und Bugs effizienter aufgedeckt werden. Mit Hilfe von Daten einer Crowdtesting Plattform haben die Autorinnen und Autoren die Nutzung ihres Werkzeugs simuliert und evaluiert.

Causal Testing: Understanding Defects' Root Causes id: a087 [24]
Innerhalb dieser Arbeit wurde ein Eclipse Plugin entwickelt, um Entwicklerinnen und Entwickler bei der Analyse von Defekten und dessen Ursachen zu unterstützen. Dabei verändert das Plugin bestehende Testfälle, sodass mehr Eingaben abgetestet werden und ein ausführlicherer Bericht bei fehlschlagenden Tests einen Bericht generiert werden kann. Dadurch können Entwicklerinnen und Entwickler sich orientieren, anhand welche Eingaben ein Testfall fehlschlägt. Das Plugin wurde mittels kontrollierten Experimenten und Defekte der Benchmark *Defects4J* getestet.

A Tale from the Trenches: Cognitive Biases and Software Development id: a654 [5]
Die Autorinnen und Autoren beschäftigen sich mit kognitiven Biasen in der Software Entwicklung, mit dessen Konsequenzen und mit Praktiken und Werkzeugen zum Entgegenwirken. Dafür haben sie eine Feldstudie durchgeführt. Sie haben 10 Entwicklerinnen und Entwickler in ihren Arbeitsalltag beobachtet und ihre Aktionen mit Artefakten dokumentiert. Außerdem haben sie 16 Interviews durchgeführt. Die beobachten Biase haben sie in 10 Gruppen kategorisiert und helfende Praktiken in 5 Gruppen eingeordnet.

Here We Go Again: Why Is It Difficult for Developers to Learn Another Programming Language? id: a691 [49]
In dieser Arbeit wird nachgegangen, ob Entwicklerinnen und Entwickler beim Lernen einer neuen Programmiersprache Schwierigkeiten haben. Dazu haben die Autorinnen und Autoren Stackoverflow Posts analysiert, um aufzudecken, wie oft Entwicklerinnen und Entwickler falsche Annahmen tätigen aufgrund ihres Vorwissen von anderen Programmiersprachen. Des Weiteren haben die Autorinnen und Autoren Inter-

[25] beispielsweise https://stephaniehohenberg.github.io/TSEK-Analyzer/?id=b535

view durchgeführt, um herauszufinden vor welchen Herausforderungen sie stehen und welche Auswirkungen bestehen.

Big Code != Big Vocabulary: Open-Vocabulary Models for Source Code id: b073 [25]
Diese Arbeit setzt sich mit *Natural Language Models* für Code auseinander, welche unter Anderem verwendet werden, um Code Vorschläge beim Programmieren zu generieren. Die Autorinnen und Autoren haben ihr eigenes Modell mit der Anpassung des *Byte-Pair Encoding* (BPE) Algorithmus erstellt und mit den aktuellen Stand der Forschung verglichen.

Unblind Your Apps: Predicting Natural-Language Labels for Mobile GUI Components by Deep Learning id: a322 [7]
Um mobile Anwendungen zugänglicher für Sehbehinderte und Blinde zu machen, müssen Labels, welche von einen ScreenReader vorgelesen werden können, an bild-basierte Komponenten hinzugefügt werden. Die Autoren und Autorinnen dieser Arbeit haben dafür eine Analyse durchgeführt, um herauszufinden, inwieweit Anwendungen ihre Komponenten labeln. Außerdem haben die Autorinnen und Autoren ein Deep Learning Modell entwickelt, um Labels für bildbasierte Aktionsknöpfe zu bestimmen.

Translating Video Recordings of Mobile App Usages into Replayable Scenarios id: a309 [3]
Innerhalb dieser Arbeit wurde ein Werkzeug entwickelt, um Videos von mobilen Anwendungen, die unter Anderem als Artefakt für Bug Berichte verwendet werden, im Textformat, in Szenario Skripten, zu übersetzen, um die Entwicklerinnen und Entwickler in ihrer Arbeit zu unterstützen. Dabei wurden Methoden zum Klassifizieren von Bildern und Erkennen von Objekten angewandt. Das Werkzeug mittels mehreren mobilen Anwendungen evaluiert und die Autorinnen und Autoren haben sich Feedback von Entwicklerinnen und Entwickler eingeholt.

Time-travel Testing of Android Apps id: a481 [10]
Diese Arbeit beschäftigt sich mit dem Testen von Android Apps, insbesondere mit dem automatischen Generieren von Eingabe- und Eventsequenzen. Die Autorinnen und Autoren fokussieren sich dabei, durch diese Sequenzen verschiedene Zustände der Anwendung zu durchlaufen und abzutesten. Dazu haben sie ein Werkzeug entwickelt, mehrere Anwendungen damit getestet und es mit bereits bestehenden Werkzeugen verglichen.

Primers or Reminders? The Effects of Existing Review Comments on Code Review id: b171 [51]
Die Autorinnen und Autoren sind der Frage nachgegangen, ob bestehende Code Review Kommentare, die weitere Code Review beeinflusst und Biase bestehen. Dazu haben sie ein kontrolliertes Experiment mit Entwicklerinnen und Entwicklern durchgeführt.

Towards the Use of the Readily Available Tests from the Release Pipeline as Performance Tests. Are We There Yet? id: b435 [9]
Diese Arbeit handelt von Tests, die in der *Release Pipeline* ausgeführt werden und zur Sicherstellung der Performanz genutzt werden können. Die Autorinnen und Autoren haben Commits von zwei Projekten analysiert, die Performanz Probleme beheben und analysiert, ob die Tests der Pipeline die Verbesserung darstellen können. Durch eine manuelle Analyse haben sie dafür relevante Faktoren aufgedeckt, die Entwicklerinnen und Entwickler helfen können.

White-box Fairness Testing through Adversarial Sampling id: a949 [67]
Die Autorinnen und Autoren setzten sich mit Fairness Testing von neuronalen Netzwerken auseinander. Dazu haben sie einen Algorithmus entwickelt, um nach diskriminierenden Instanzen zu suchen. Diesen haben sie mit den bestehenden Stand der Forschung verglichen.

Taxonomy of Real Faults in Deep Learning Systems id: b110 [20]
Die Autorinnen und Autoren haben mittels einer manuellen Analyse von Github und Stackoverflow Artefakten Fehler in Deep Learning Systemen kategorisiert und dadurch eine Taxonomie mit fünf Hauptkategorien erstellt. Durch strukturierte Interviews konnte sie diese erweitern und durch eine Umfragen haben sie die Taxonomie validiert.

An Empirical Study on Program Failures of Deep Learning Jobs id: b159 [68]
Innerhalb dieser Studie wurde das Fehlschlagen von Deep Learning Jobs auf Deep Learning Plattformen untersucht. Dazu wurden Fehlermeldungen manuell analysiert und kategorisiert. Es wurden Ursachen, sowie Lösungen vorgestellt. Die Autorinnen und Autoren haben außerdem noch Interviews mit Entwicklerinnen und Entwicklern durchgeführt.

HyDiff: Hybrid Differential Software Analysis id: b273 [38]
In dieser Arbeit wird ein Werkzeug zur differentialen Software Analyse vorgestellt, welche verschiedene Ansätze kombiniert: *Feedback-Directed Greybox Fuzzing* und

Symbolic Execution. Durch Experimente wurden die drei genannten Anwendungsfelder evaluiert: die Analyse von Side Channels, Regression Testing und die Analyse von neuronalen Netzwerken bezüglich der Robustheit.

SLEMI: Equivalence Modulo Input (EMI) Based Mutation of CPS Models for Finding Compiler Bugs in Simulink id: a335 [8]
Die Autorinnen und Autoren haben eine Technik entworfen, um Differential Testing in Cyber Physical Systemen, wie Matlab oder Simulink von MathWorks, zu unterstützen und somit auch das Aufdecken von Bugs. Mittels Experimenten und generierten Modellen haben sie Funktionalitäten von Simulink evaluiert. Außerdem haben Sie ihre Ergebnisse mit einem bereits bestehen Werkzeug verglichen.

Efficient Generation of Error-Inducing Floating-Point Inputs via Symbolic Execution id: b261 [16]
Diese Arbeit beschäftigt sich mit Fehlern, die durch Aufrunden von Floating-Point Eingaben geschehen. Dabei entwickeln sie ein Werkzeug, welches mittels *Symbolic Execution* Testeingaben generiert, um die Korrektheit, Genauigkeit, sowie die Stabilität von numerischen Programmen zu garantieren. Dieses Werkzeug haben die Autorinnen und Autoren anhand mehrerer Programme evaluiert und dadurch mehrere Bugs aufgedeckt. Außerdem haben sie einen States-of-Arts Ansatz verfolgt.

Ankou: Guiding Grey-box Fuzzing towards Combinatorial Difference id: b024 [34]
In dieser Arbeit wurde ein Werkzeug zum Aufdecken von Bugs mittels mittels *Grey-box Fuzzing* entwickelt. Dazu haben sie die Fitness Funktion des Fuzzers angepasst, sodass diese informativer ist, und nicht nur auf die Codeabdeckung beim Testen achtet. Das Werkzeug wurde mittels Experimenten evaluiert und mit zwei bereits bestehenden Werkzeugen verglichen.

On the Efficiency of Test Suite based Program Repair id: a615 [32]
Die Autorinnen und Autoren evaluieren und vergleichen in dieser Studie die Effizienz von 16*Automated Program Repair* Systeme. Dabei konzentrieren sie sich auf die generierten Patches mit denen fehlerhafte Code gefixt wird. Mittels Java Programmen der Defect4J Benchmark wurden zahlreiche Experimente durchgeführt.

Quickly Generating Diverse Valid Test Inputs with Reinforcement Learning id: b410 [47]
In dieser Arbeit wurde ein Werkzeug entwickelt, welches mittels *Reinforcement Learning* Eingaben für Test generiert. Dabei wurde ein *black-box* Ansatz verfolgt. Mittels

Benchmark wurde das Werkzeug evaluiert und mit der *States-of-Arts* verglichen.

Practical Fault Detection in Puppet Programs id: a026 [50]
Diese Arbeit beschäftigt sich mit Puppet, ein Werkzeug zur Konfiguration von Computer Systemen. Innerhalb der Arbeit wurde ein Ansatz entwickelt um die Interaktionen und Abhängigkeiten zwischen Puppet Ressourcen und die Systemaufrufe eines Puppet Programms zu analysieren und um somit Fehler aufzudecken. Der Ansatz wurde anhand 33 Puppet Modulen und Performanz Benchmarks evaluiert.

Gap between Theory and Practice : An Empirical Study of Security Patches in Solidity id: a542 [21]
In dieser Arbeit untersuchen die Autorinnen und Autoren, inwieweit die entwickelten Analyse Werkzeuge und Frameworks zum Aufdecken von Sicherheitslücken in Ethereum Smart Contracts, angewendet werden. Dabei haben sie herausgefunden, dass viele aufgedeckte Sicherheitslücken noch nicht im Solidity Compiler integriert wurden und zusätzlich dazu benutzen viele Entwicklerinnen und Entwickler ältere Version von dem Solidity Compiler.

Debugging Inputs id: a075 [27]
Die Autorinnen und Autoren dieser Arbeit haben ein Werkzeug entwickelt um Eingaben, die zu einem Programmfehler führen, zu regenerieren und reparieren, um somit unvollständige oder beschädigte Daten zu retten und eventuell trotzdem verarbeiten zu können. Das Werkzeug wurde anhand mehreren Programmen und mit Eingaben, die in ihrer Größe und in ihren Formaten variieren, evaluiert.

Empirical Review of Automated Analysis Tools on 47,587 Ethereum Smart Contracts id: a530 [11]
In dieser empirischen Studie werden neun automatisierte Analyse Werkzeuge für das Aufdecken von Schwachstellen in Ethereum Smart Contracts evaluiert und verglichen. Dazu wurde sich ein Überblick über den aktuellen Entwicklungsstand solcher Werkzeuge geschaffen und mit Hilfe eines Frameworks wurden die Werkzeuge ausgeführt und die Verträge analysiert.

Automatically Testing String Solvers id: b459 [4]
Diese Arbeit beschäftigt sich mit der Korrektheit von *String Solvers*. Dazu wurde ein Ansatz entwickelt, welcher Formeln generiert, um diese Solver zu testen. Der Ansatz wurde mit drei SMT-Solver und einen Automaten-basierten Solver evaluiert und mit den existieren *Fuzzing* Techniken verglichen.

MemLock: Memory Usage Guided Fuzzing id: a765 [59]
Die Autorinnen und Autoren entwickelten in dieser Arbeit eine Fuzzing Technik, um große Eingaben zu generieren, um somit *memory consumption bugs* zu triggern und Sicherheitslücken aufzudecken. Diese Technik wurde anhand mehrerer Programme evaluiert und mit anderen Fuzzing Techniken, die sich mehr auf die Codeabdeckung konzentrieren verglichen.

Symbolic Verification of Message Passing Interface Programs id: b248 [65]
In dieser Arbeit wurde ein Werkzeug entwickelt um Programme, die das *Message Passing Interface* (MPI) Paradigma verwenden zu verifizieren. Darunter zählen unter Anderem *High Performance Computing* Anwendungen. Das Werkzeug nutzt die Techniken *Symbolic Execution* und *Model Checking* und wurde anhand MPI Programmen evaluiert.

SAVER: Scalable, Precise, and Safe Memory-Error Repair id: a271 [17]
Die Autorinnen und Autoren dieser Arbeit haben eine Technik entwickelt um *Memory Bugs* zu reparieren. Dabei wird durch eine statische Analyse das Verhalten eines Programms hinsichtlich des Memory Heap's in einen Flussgraphen dargestellt. Die Technik wurde auf 11 C Programmen angewandt und mit einen anderen Werkzeug verglichen.

Neurological Divide: An fMRI Study of Prose and Code Writing id: a678 [28]
In dieser Studie werden die kognitiven Prozesse während dem Schreiben von Code und während dem Schreiben von Prosa verglichen. Dazu wurden Brain Scans mit Studierenden durchgeführt und verglichen, inwieweit sich die Gehirnaktivität bei den Ausführen der Aufgaben ähnelt.

TRADER: Trace Divergence Analysis and Embedding Regulation for Debugging Recurrent Neural Networks id: a986 [53]
Die Arbeit beschäftigt sich mit textuellen Eingaben für neuronale Netzwerke, in denen numerische Werte eingebettet sind. Durch eine dynamische Diagnose wurden diese Einbettungen in Bezug auf die Model Performanz analysiert, und die Einbettungen korrigiert um die Model Genauigkeit zu erhöhen.

Testing DNN Image Classifiers for Confusion Bias Errors id: b122 [54]
Innerhalb dieser Arbeit wurden neurale Netzwerke, welche zur Klassifikation von Bilder genutzt werden, auf ihre Robustheit getestet. Dazu wurde getestet, inwieweit die Eigenschaften dieser Klassen verletzt wurden und somit Fehler in der Klassifikation vorhanden sind. Dazu wurde eine Werkzeug entwickelt, welches an 6 Daten-

sätzen und 8 Modellen angewandt und evaluiert wurde.

Repairing Deep Neural Networks: Fix Patterns and Challenges id: b135 [22]
Die Autorinnen und Autoren dieser Arbeit beschäftigen sich mit den Mustern von Bugfix Lösungen für neurale Netzwerke und die Herausforderungen diesbezüglich. Dazu wurden Lösungen von Stackoverflow Posts und GitHub Code analysiert und mittels *OpenCoding* klassifiziert. Zusätzlich zu dieser Studie wurde eine Benchmark veröffentlicht.

Lazy Product Discovery in Huge Configuration Spaces id: b509 [31]
Diese Arbeit konzentriert sich auf hoch konfigurierbare Systemen, deren Subsysteme voneinander abhängige Konfigurationsoptionen haben können. Dabei wurde eine Technik entwickelt, um gültige Konfigurationen aufzudecken. Dazu wird der Konfigurationsbereich durch ein Modell abgebildet. Die Technik wurde innerhalb eines Experimentes mit einen industriellen System evaluiert.

Managing data constraints in database-backed web applications id: b098 [64]
Die Autorinnen und Autoren dieser Arbeit haben eine Studie durchgeführt um unter Anderem zu analysieren, wie die einer Anwendung eingeschränkt werden, wie diese Einschränkungen ausgedrückt und inwieweit diese verletzt werden. Dazu haben sie 12 Ruby on Rails Anwendungen analysiert und aus deren Resultate haben sie ein Werkzeug, welches Anti-Patterns automatisch aufgedeckt, entwickelt.

Improving Data Scientist Efficiency with Provenance id: b086 [19]
Innerhalb dieser Arbeit wurde ein Werkzeug entwickelt, welches bei dem Ausführen von Skripten, welche zur Datenanalyse genutzt werden, unterstützt. Das Werkzeug generiert dabei ein vereinfachtes Skript. Es wird nicht das gesamte Skript ausgeführt, sondern nur der nötigste Code, sodass das Anpassen von Parametern oder das Debuggen von Skripten erleichtert wird. Mittels zwei verschiedenen Nutzerstudien wurde das Werkzeug von der Zielgruppe evaluiert.

On the Recall of Static Call Graph Construction in Practice id: b049 [52]
Das Paper beschäftigt sich mit der statischen Konstruierung von Call Graphen in Objekt-orientierten Programmiersprachen mit dem Ergebnis das hybride Lösungen notwendig sind, eine Kombination aus statischen und dynamischen Lösungen, um ohne Performanceprobleme die dynamischen Eigenschaften einer Programmiersprache abdecken zu können und modellieren zu können.

Watchman: Monitoring Dependency Conflicts for Python Library Ecosystem id: a125 [58]
Dieses Paper handelt von Dependency Konflikten in Python Projekten, welche abhängig von Bibliotheken des third-party software repository *Python Package Index (PyPi)* sind. Bei dem Importieren von Bibliotheken, die wiederum andere Bibliotheken nutzen, kann es zu Versionskonflikten kommen, welche *build failure*, *runtime error* oder semantische Inkonsistenzen hervorrufen können.
Um die Ursachen von solchen Fehlern und Inkonsistenzen diagnostizieren zu können und Dependency Konflikte zu entdecken, haben die Autorinnen und Autoren ein Werkzeug entwickelt. Des Weiteren wurden Dependency Konflikte manuell analysiert und herausgefunden, wie diese entstehen.

Revealing Injection Vulnerabilities by Leveraging Existing Tests id: a284 [18]
Die Verfasserinnen und Verfasser beschäftigen sich mit dem Aufdecken von Code Injection Attacken und mit den *Vulnerabilities* von Applikationen. Dabei haben sie ein Werkzeug entwickelt, welches den Informationsfluss mittels dynamischen Taint Tracking beobachtet. Um den Overhead während Runtime zu vermeiden, wird das Tool nicht während Runtime ausgeführt, wie bei typischen Taint Tracking Frameworks, sondern vor dem *Deployment* bei dem Ausführen von Tests. Dazu werden White-Box-Tests generiert, um geeignete, bereits bestehende, funktionale Tests wiederzuverwenden.
Mittels einer Benchmark wurde dieser Ansatz mit einem anderen Werkzeug verglichen.

Heaps'n Leaks: How Heap Snapshots Improve Android Taint Analysis id: b061 [2]
Die Verfasserinnen und Verfasser dieser Arbeit haben sich zum Ziel die Taint Analyse für Android Applikationen zu verbessern, um Sicherheitslücken, die durch Bugs entstanden sind, zu schließen. Die Taint Analyse verfolgt den Informationsfluss von Eingabedaten der Nutzerin bzw. des Nutzers, welche unter Anderem sensitiv sein können, wie Bankinformationen oder private Bilder. Mittels Heap Snapshots, welche die gespeicherten Objekte, deren Werte und Beziehungen zu einen bestimmten Zeitpunkt repräsentiert, möchten die Verfasserinnen und Verfasser mehr Einblick in die Taint Analyse erhalten.
Dazu wurde ein bereits bestehendes Werkzeug erweitert und evaluiert. Des Weiteren wurde eine Benchmark erstellt und Experimente eines anderen Papers reproduziert, um weitere Forschungsfragen zu beantworten.

Verifying Object Construction id: b447 [26]
In dieser Arbeit haben die Autorinnen und Autoren eine Werkzeug entwickelt, um die Objekt Konstruktion mittels dem *Builder* Lombok zu verbessern, sodass diese korrekt verwendet werden und Objekte mit gültigen Werte konstruiert werden. Dadurch sollen Sicherheitslücken oder Null-Pointer Verletzungen vermieden werden. Das Werkzeug haben sie mit einer Nutzerstudie und einer anschließenden Umfrage getestet.

ComboDroid: Generating High-Quality Test Inputs for Android Apps via Use Case Combinations id: a469 [56]
Innerhalb dieser Arbeit wurde ein Test Framework entwickelt, um Funktionalitäten in Android Anwendungen semi-automatisch zu testen, für die große Testeingaben nötig sind. Dazu wurde die Testeingabe in zerlegt, um unabhängige Anwendungsfälle darzustellen bzw. App Funktionalitäten abdecken. Diese Anwendungsfälle können mit verschiedenen Werte kombiniert werden. Der Ansatz wurde mit anderen Techniken verglichen.

Near-Duplicate Detection in Web App Model Inference id: a186 [63]
Diese Arbeit beschäftigt sich mit dem Generieren von Tests für Webanwendungen durch das Ableiten von Modellen, die diese Webanwendung darstellen. Um die Herausforderungen dabei zu untersuchen, haben die Autorinnen und Autoren 10 verschiedene Techniken verglichen.

Tailoring Programs for Static Analysis via Program Transformation id: a824 [55]
Die Arbeit untersucht Werkzeuge, die Code statisch analysiert, um Bugs zu entdecken. Es wurde eine Technik entwickelt, die den Code temporär verändert, sodass falsche Fehlermeldungen dieser Werkzeuge umgangen werden. Mittels einen regel-basierten Ansatz und deklarativen Templates, werden Code Muster, die problematisch für die Analyse Werkzeuge sind transformiert, sodass der Bericht der Analysewerkzeuge reduziert wird. Die Technik wurde mit 5 verschiedenen Werkzeugen und Programmen in verschiedenen Programmiersprachen evaluiert.

Gang of Eight: A Defect Taxonomy for Infrastructure as Code Scripts id: a752 [46]
Die Autorinnen und Autoren dieser Arbeit haben sich mit den Defekten in Puppet Skripten beschäftigt. Dazu haben sie Commits, die sich auf Defekte beziehen, analysiert, um Defekte zu klassifizieren und eine Taxonomie zu bilden. Mittels einer Umfrage haben sie die Taxonomie, die aus 8 Kategorien besteht, evaluiert und im Anschluss ein Werkzeug entwickelt, um Defekte in Commits automatisch zu identifizieren und klassifizieren.

Recognizing Developers' Emotions while Programming id: a666 [15]
Innerhalb dieser Arbeit wurden die Emotionen beim Programmieren analysiert. Dazu wurde Experimente durchgeführt, in denen die Testpersonen Sensoren tragen, und Machine Learning angewandt, um Emotionen aufzudecken und zu klassifizieren. Dabei haben die Autorinnen und Autoren analysiert, was die positiven und negativen Emotion auslöst und wie mit den negative Emotionen umgegangen werden kann.

HeteroRefactor: Refactoring for Heterogeneous Computing with FPGA id: a493 [29]
Die Autorinnen und Autoren dieser Arbeit haben eine Technik entwickelt, welche C/C++ Programme transformiert, sodass diese für FPGA's, reprogrammierbare Hardware, genutzt werden können. Dabei wurde drei verschiedene Refactoring Maßnahmen bezüglich rekursive Datenstrukturen, Integer und Floating Points angewandt. Dadurch werden die Ressourcen des FPGA's weniger und somit optimaler genutzt.

Multiple-Entry Testing of Android Applications by Constructing Activity Launching Contexts id: a457 [62]
In dieser Arbeit wird ein Framework vorgestellt, um die GUI von Android Anwendungen explorativ zu testen. Dazu werden verschiedene Analysen durchgeführt, um Modelle und Kontexte zu entwickeln. Mittels dieser Kontexte, können Aktivitäten gestarten werden, sodass Funktionalitäten der Anwendung getestet werden können, ohne dass lange Sequenzen von Events und Benutzereingaben durchgeführt werden müssen. Das Framework wurde anhand 20 Anwendungen evaluiert und mit einem anderen Werkzeug verglichen.

CPC: Automatically Classifying and Propagating Natural Language Comments via Program Analysis id: b359 [66]
Innerhalb dieses Papers wurde ein Prototyp entwickelt, der Programme automatisch analysiert und mit Hilfe eines Machine Learning Algorithmus Kommentare klassifiziert und neue Kommentare für Methoden generiert. Die Nützlichkeit dieser Kommentare wurde durch eine Nutzerstudie evaluiert. Der Prototyp könnte bei dem Entdecken von Bugs helfen. Dazu wurden generierte Kommentare, welche *Exceptions* beschreiben, mittels eines Skripts extrahiert und im Anschluss analysiert.

Understanding the Automated Parameter Optimization on Transfer Learning for Cross-Project Defect Prediction: An Empirical Study id: a566 [30]
Um ein Modell zu entwickeln, welches Defekte in einem Projekt aufgedeckt, werden mit Hilfe von *Cross-Project Defect Prediction* (CPDP) Techniken, Daten von anderen Projekten genutzt. Die Autorinnen und Autoren dieser Studie beschäftigen sich, in wie weit das automatische Optimieren von Parametern diese Techniken verbessern kann. Dazu wurden 20 Projekte und 69 CPDP Techniken genutzt um Modelle zu bauen und zu analysieren.

How Does Misconfiguration of Analytic Services Compromise Mobile Privacy? id: b572 [69]

Dieses Paper beschäftigt sich mit der Privatsphäre und den Datenschutz innerhalb mobile Anwendungen, in denen analytische Services eingebunden sind, um das Nutzerverhalten zu analysieren, Bugs aufzudecken und personalisierte Werbung zu schalten. Dabei haben die Autorinnen und Autoren mittels eines entwickelten Werkzeugs mobile Anwendungen analysiert, deren Konfigurationen von analytischen Services untersucht, und sind der Frage nachgegangen, inwieweit fehlerhafte Einstellungen zu Datenschutzverstößen führen können.

Testing File System Implementations on Layered Models id: b483 [6]
Die Autorinnen und Autoren beschäftigen sich mit dem Testen von dem Dateisystem Linux. Dazu entwickeln sie eine Methode um den Sequenzen von Systemaufrufen als Input für solche Tests zu generieren. Das daraus resultierende Werkzeug wurde mittels Experimenten evaluiert.

Burn After Reading: A Shadow Stack with Microsecond-level Runtime Rerandomization for Protecting Return Addresses id: a258 [70]
Das Paper beschäftigt sich mit der Sicherheit von C/C++ Programmen, insbesondere mit *Code Reuse* Attacken durch *Return-orientated Programming*. Dazu wurde ein Mechanismus entwickelt, welcher verschiedene Techniken vereint, um *Return Addresses* zu schützen, sodass diese nicht überschrieben und umgeleitet werden können. Der entwickelte Mechanismus wurde mittels C/C++ Benchmarks mit einen bereits bestehenden Mechanismus verglichen und evaluiert.

Mitigating Turnover with Code Review Recommendation: Balancing Expertise, Workload, and Knowledge Distribution id: b183 [36]
Die Autorinnen und Autoren dieses Paper haben ein Werkzeug entwickelt, um die Code Review von Code zu unterstützen, sodass Wissen unter den Entwicklerinnen und Entwicklern verteilt wird. Das Werkzeug wurde mit Reviews von bereits vergangenen Projekten simuliert angewandt und evaluiert.

A Study on the Prevalence of Human Values in Software Engineering Publications, 2015 – 2018 id: a409 [43]
Innerhalb dieses Papers wurde eine Pilotstudie durchgeführt um herauszufinden, inwiefern zu menschlichen Werten in der Software Engineering Welt geforscht wird. Dazu wurde Literatur von verschiedenen führenden Zeitschriften und Konferenzen klassifiziert und analysiert, darunter auch die ICSE. Die Autorinnen und Autoren haben beobachtet, dass innerhalb der ausgewählten Literatur sich nur ein geringer Anteil mit menschlichen Werten beschäftigt (16%) und davon 41% mit *security*, was sie als problematisch ansehen.

POSIT: Simultaneously Tagging Natural and Programming Languages id: b348 [41]

Die Autoren und Autorinnen dieses Paper haben eine Methode entwickelt um Texte von Entwicklerinnen und Entwicklern, die sowohl natürliche Sprache, als auch Code enthalten, zu taggen. Dazu wurde eine neuronales Netzwerk mittels Stackoverflow Posts und Code Kommentaren trainiert, evaluiert und mit dem aktuellen Stand der Forschung verglichen.

Detection of Hidden Feature Requests from Massive Chat Messages via Deep Siamese Network id: a641 [48]
In diesem Paper geht es um die Online Kommunikation von Entwicklerinnen und Entwicklern. Es wurde ein Ansatz vorgestellt, um aus deren Chatnachrichten potentielle Features zu entdecken um dadurch das Requirements Engineering zu unterstützen. Dazu wurde ein Werkzeug entwickelt, welches Chat Dialoge miteinander verglichen und klassifiziert, anstatt die Nachrichten satzweise zu analysieren. Im Anschluss wurde das Werkzeug anhand drei Projekte mit anderen Werkzeugen verglichen.

SLACC: Simion-based Language Agnostic Code Clones id: a210 [35]
Die Autorinnen und Autoren dieses Papers haben einen Framework zur dynamischen Analyse von Code entwickelt Framework um Code Clones aufzudecken. Sprachübergreifend wird Code in Java und Python anhand des Eingabe-/Ausgabe Verhalten verglichen. Das Framework wurde mit einen bereits bestehenden Werkzeug für Code Clone Detection in Java verglichen.

ReluDiff: Differential Verification of Deep Neural Networks id: a714 [42]
Innerhalb dieses Paper wurde eine Methode entwickelt zur differentialen Verifikation von neuronalen Netzwerken, um formal sicher zu stellen, dass die Kompression von Netzwerken nicht zu Unstimmigkeiten führen. Dabei lag der Fokus auf miteinander zusammenhängende Netzwerke. Mittels Experimenten und einer Benchmark von 49 Netzwerken aus 3 verschiedenen Domänen wurde diese Methode mit bereits bestehenden Werkzeugen verglichen.

How to Not Get Rich: An Empirical Study of Donations in Open Source id: b209 [40]
Diese empirische Studie untersucht OpenSource Projekte innerhalb des GitHub und npm Korpus, die nach Finanzierung fragen. Dazu wurden öffentlich zugängliche Ressourcen manuell analysiert. Daraus wurden mehrere Schlussfolgerungen gezogen und weitere Forschungsfragen bezüglich der effektiven Finanzierung aufgeworfen.

Securing UnSafe Rust Programs with XRust id: a234 [33]
Die Autorinnen und Autoren des Papers stellen eine Methode vor um Sicherheitslücken innerhalb der Programmiersprache Rust zu schließen, indem der Speicherzugriff von sogenannten *unsafe Rust* geschützt wird. Zur Evaluierung wurden Experimente mit Rust Anwendungen und Standardbibliotheken durchgeführt.

7.3 Test- und Fragebogen zum Usability Test und zum semi-strukturierten Interview

Im Folgendem ist der Test- und Fragebogen zur Unterstützung der Usability Tests und den semi-strukturierten Interviews einsehbar, welche zur Evaluierung des Prototyps durchgeführt wurden und in Kapitel 5 Sektion 5.3 genauer erläutert werden.

Test- und Fragebogen

Name: Geschlecht: w/m/d
Position: Berufserfahrung: n+ Abschluss: MSc/BSc 20__
Bist du damit einverstanden, dass ich Zitate anonym in meiner Masterarbeit einbringe?

Aufgabe 1 Recherche	
Aufgabe 2 Datenanalyse	
Feedback zum Tool	Was fandest du gut / schlecht am Prototypen? Was würdest du verbessern? Was ist dir aufgefallen?
Fragen	1) Informierst du dich regelmäßig über den aktuellen Stand von SE? Liest du akademische Paper? 2) Welche Medien nutzt du? (Blogs, Zeitschriften, akademische Paper, Google News Recommender, Empfehlungen von Kollegen, Reddit …) 3) Besuchst du Konferenzen, wo akademische Paper vorgestellt werden? 4) Wie stehst du in Kontakt zu der Uni/ der akademischen Welt? 5) Wie siehst du das Verhältnis zwischen der akademischen und der industriellen Welt? Und in Bezug auf Wissenstransfer? 6) Welche Datenaufbereitung, -visualisierung oder Formate würdest du bevorzugen, um einen Wissenstransfer / Austausch zu fördern? 7) Würde dir eine Taxonomie von den aktuellen Beiträgen in der akademischen Welt des SE interessieren? (Vergleich Technology Radar)

7.4 Anschreiben und Fragebogen zu den semi-strukturierten Interviews mit den ICSE Program Board

Im Folgenden ist das Anschreiben an die Gutachterinnen und Gutachter der ICSE 2020 zur Teilnahme an einem Interview einsehbar, sowie der Fragebogen, der zur Unterstützung des semi-strukturierten Interviews diente.

Anschreiben der Gutachterinnen und Gutachter

Subject: Interview about the ICSE 2020

Dear Prof. Dr. <Name>,

My name is Stephanie Hohenberg and I am a student of the Free University Berlin in Germany, completing my master degree of computer science. Currently I am writing my master thesis 'First steps towards a taxonomy of software engineering contexts'. You can find the abstract in the attachment.

For this matter, I conducted a literature study reviewing 59 technical papers of the ICSE 2020. I am planning to conduct interviews to not just gain better understanding in the process, how papers are reviewed, by which criteria papers get judged and how papers get accepted, but also to evaluate whether my proposed prototype and the resulting taxonomy of the thesis could be useful and relevant in this process.

You as a member of the program board for the ICSE 2020, would be a perfect fit for my planned interview and I would be glad, if you could spend 30 minutes of your time to answer my questions. In case you would prefer to answer written questions by mail instead of conducting an interview online, I am keen to send you my questions instead.

Thank you for your effort and time in advance.

Sincerely,
Stephanie Hohenberg

Fragebogen zum semi-strukturierten Interview

1 General Understanding of the Submission Process

3 Questions about the Submission Process to gain a big picture of the ICSE.

2 Paper Review Process

Question 2-1
Do you use an **evaluation sheet or a tool to support** you in the paper review?

Question 2-2
How are you proceeding in the paper review? Do you have a **special reading technique**? Is every paper review using their own reading technique, their own method to review a paper or is there a standardized procedure or guidelines by the ICSE?

Question 2-3
How is the **score** of a paper calculated? On what scale (and unit) is the score represented f.e. 1-100points?

Question 2-4
Following this LINK, the review criteria is mentioned. It got mentioned that the **type of contribution matters**.
Could you specify whether and how you are paying attention to the type of contribution, while you are reviewing a paper?
Have you noticed or do you assume, any correlations between a specific review criteria or the quality of a paper and the type of contribution?

Question 2-5
Following this LINK, the review criteria is mentioned. One of the review criteria is the **soundness**. "Soundness: The extent to which the paper's contributions are supported by rigorous application of appropriate research methods" I think this criteria is fitting best to how I conduct my literature study and how I review papers.

In my thesis, I am evaluating how authors are writing about the context, in which their work is moving or in which their results are applicable or generalizable.
While I conducted a literature study I got the impression that authors are not explicitly describing what their context is. Do you think this might be a problem? Do you think it is desirable to **explicitly describe the context** of a paper?

Question 2-6
I am still in the data analysis of my conducted literature study, but I noticed the tendency to write about contexts related to the application in the beginning of the paper to introduce the topic and sometimes as well to strengthen the relevance.
Do you think it is desirable to describe this kind of context more explicitly?
Do you wish there would be a writing guideline standard or a tool with a taxonomy, so you can see the application contexts of a paper?

Fragebogen zum semi-strukturierten Interview

3 Segmentation

To analyse the relevance of my thesis, I segmented the field of SE into different roles and the biggest segments and the first division is between academic and industrial roles. In my thesis I evaluate these roles and how applicable my thesis results, especially the taxonomy are for the different roles.

Question 3-1 relationship
How do you perceive the relation between the academic and the industrial world? Can you even divide the field of SE strictly into two worlds?

Question 3-2
Do you see a responsibility in the academic world or in the industrial world to strengthen their relations and exchange? Do you see the responsibility in the individual or in the system?

Question 3-3
Do you think a taxonomie from the contexts of the current contribution would be relevant in the relationship between the academic and industrial world?

4 Relevance of the Taxonomy

Question 4-1
Do you think a taxonomie from the contexts of the ICSE could be helpful for you in your role as a paper reviewer in the paper review step?
Do you see any advantages in the decision process?

Question 4-2
Could a taxonomy of contexts support the authors to submit their papers?

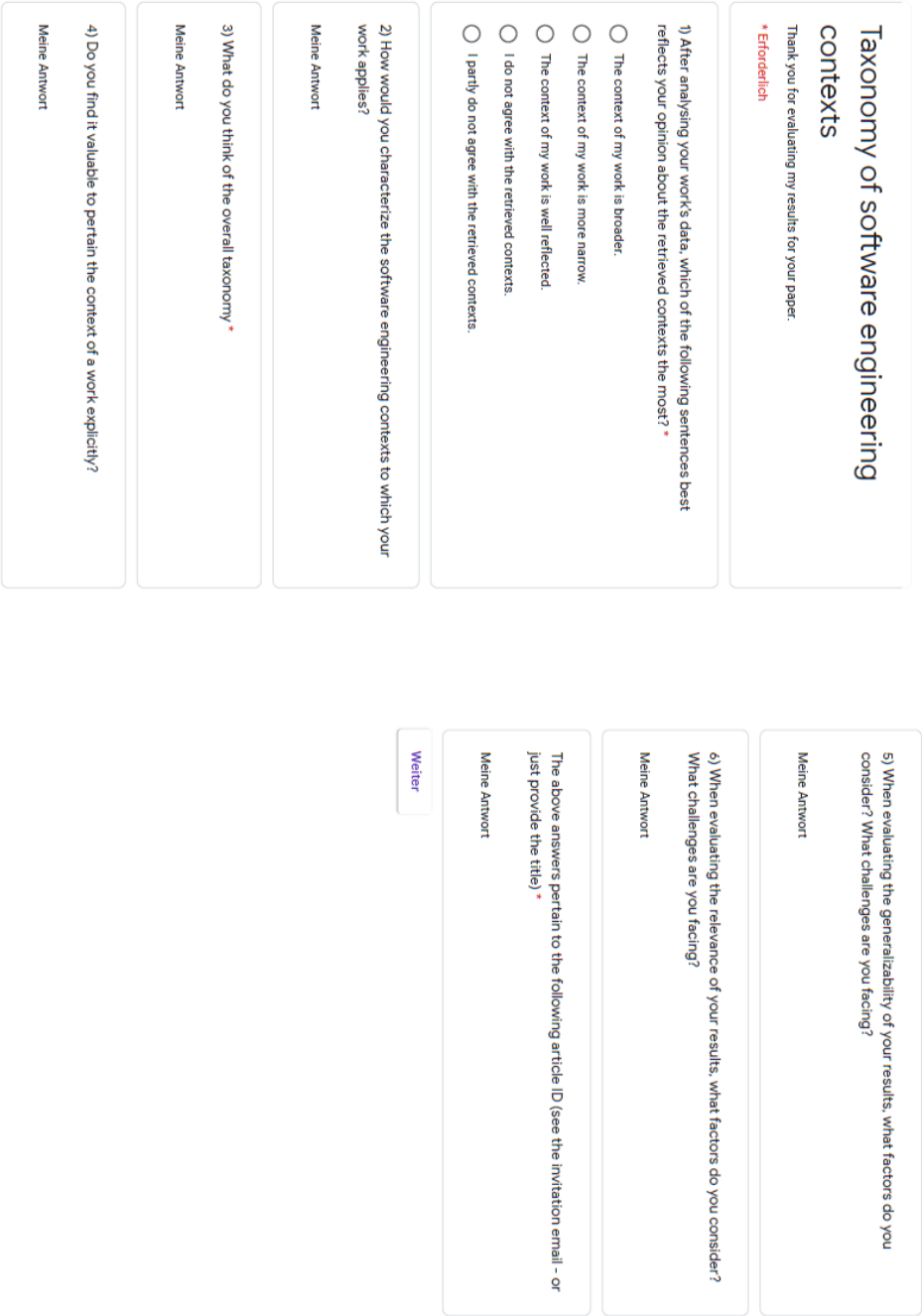
Question 4-3
Could a taxonomy of contexts support the communication between reviewers within the programming committee?

Question 4-4
Could a taxonomy of contexts support you in the assessment of a specific review criteria ?
Which review criteria would a taxonomy support the most?

- Soundness, Significance, Novelty, Verifiability, Presentation

7.5 Anschreiben und Umfrage an die Autorinnen und Autoren

Im Folgenden ist das Anschreiben an die Autorinnen und Autoren der gesichteten Arbeiten zur Teilnahme an eine Umfrage, sowie die Umfrage selbst einsehbar.



Taxonomy of software engineering contexts

Thank you for evaluating my results for your paper.

* Erforderlich

1) After analysing your work's data, which of the following sentences best reflects your opinion about the retrieved contexts the most? *

- The context of my work is broader.
- The context of my work is more narrow.
- The context of my work is well reflected.
- I do not agree with the retrieved contexts.
- I partly do not agree with the retrieved contexts.

2) How would you characterize the software engineering contexts to which your work applies?

Meine Antwort

3) What do you think of the overall taxonomy *

Meine Antwort

4) Do you find it valuable to pertain the context of a work explicitly?

Meine Antwort

5) When evaluating the generalizability of your results, what factors do you consider? What challenges are you facing?

Meine Antwort

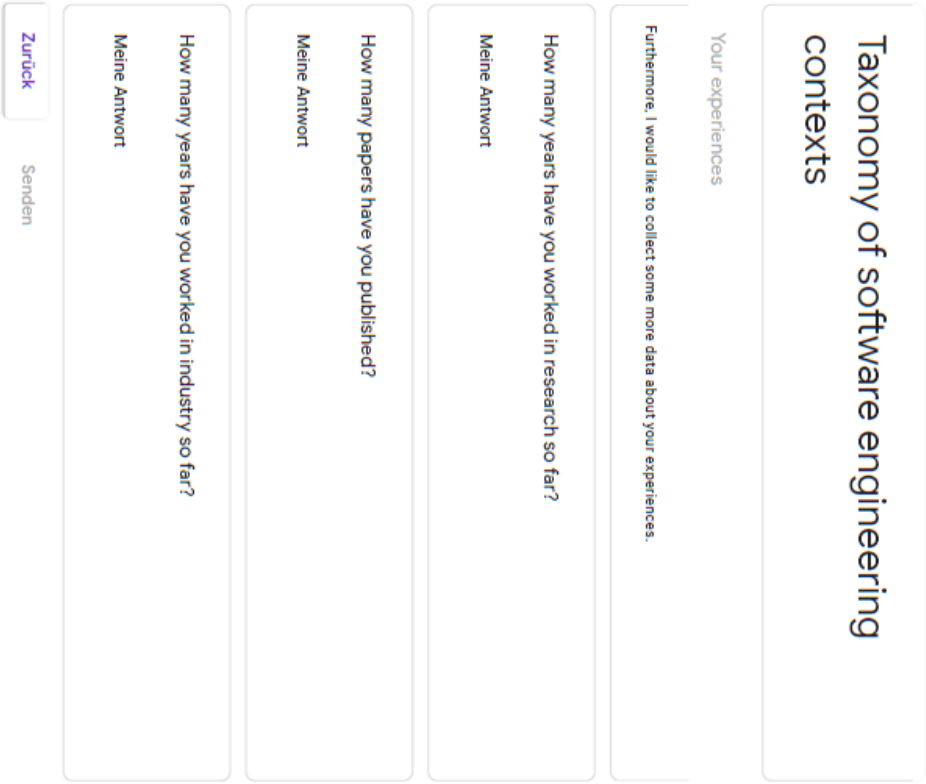
6) When evaluating the relevance of your results, what factors do you consider? What challenges are you facing?

Meine Antwort

The above answers pertain to the following article ID (see the invitation email - or just provide the title) *

Meine Antwort

Weiter



Anschreiben der Autorinnen und Autoren

Subject: First steps towards a taxonomy of software engineering contexts

Dear authors of the paper '<title of the paper>',

For my master thesis 'First steps towards a taxonomy of software engineering contexts', I conducted a literature study of ICSE 2020, where you published a technical paper. I read your work several times, evaluated it, retrieved its "context" and integrated it in a taxonomy.

Please support me by checking whether I classified your work appropriately.

In the pdf attached you can view the data I retrieved from your work. I would be very thankful if you could analyze it and answer 6 questions about it, which you can find following this link: <link> (until 2021-07-10). Please remember to enter the ID of your work in the questionnaire: <ID>

If you are interested in more background information about my thesis, the abstract is attached to this mail as well.

Thank you for your time and effort in advance.

Sincerely,
Stephanie Hohenberg
Freie Universität Berlin

8 Literatur

[1] **ACM**. *ACM Computing Classification System*. https://dl.acm.org/ccs. [Online; accessed 23-August-2021]. 2021.

[2] **Manuel Benz, Erik Krogh Kristensen, Linghui Luo, Nataniel P. Borges, Eric Bodden, and Andreas Zeller**. "Heaps'n Leaks: How Heap Snapshots Improve Android Taint Analysis". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1061–1072. ISBN: 9781450371216. DOI: 10.1145/3377811.3380438. URL: https://doi.org/10.1145/3377811.3380438.

[3] **Carlos Bernal-Cárdenas, Nathan Cooper, Kevin Moran, Oscar Chaparro, Andrian Marcus, and Denys Poshyvanyk**. "Translating Video Recordings of Mobile App Usages into Replayable Scenarios". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 309–321. ISBN: 9781450371216. DOI: 10.1145/3377811.3380328. URL: https://doi.org/10.1145/3377811.3380328.

[4] **Alexandra Bugariu and Peter Müller**. "Automatically Testing String Solvers". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1459–1470. ISBN: 9781450371216. DOI: 10.1145/3377811.3380398. URL: https://doi.org/10.1145/3377811.3380398.

[5] **Souti Chattopadhyay, Nicholas Nelson, Audrey Au, Natalia Morales, Christopher Sanchez, Rahul Pandita, and Anita Sarma**. "A Tale from the Trenches: Cognitive Biases and Software Development". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 654–665. ISBN: 9781450371216. DOI: 10.1145/3377811.3380330. URL: https://doi.org/10.1145/3377811.3380330.

[6] **Dongjie Chen, Yanyan Jiang, Chang Xu, Xiaoxing Ma, and Jian Lu**. "Testing File System Implementations on Layered Models". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1483–1495. ISBN: 9781450371216. DOI: 10.1145/3377811.3380350. URL: https://doi.org/10.1145/3377811.3380350.

[7] **Jieshan Chen, Chunyang Chen, Zhenchang Xing, Xiwei Xu, Liming Zhu, Guoqiang Li, and Jinshui Wang**. "Unblind Your Apps: Predicting Natural-Language Labels for Mobile GUI Components by Deep Learning". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 322–334. ISBN: 9781450371216. DOI: 10.1145/3377811.3380327. URL: https://doi.org/10.1145/3377811.3380327.

[8] **Shafiul Azam Chowdhury, Sohil Lal Shrestha, Taylor T. Johnson, and Christoph Csallner**. “SLEMI: Equivalence modulo Input (EMI) Based Mutation of CPS Models for Finding Compiler Bugs in Simulink”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 335–346. ISBN: 9781450371216. DOI: 10.1145/3377811.3380381. URL: https://doi.org/10.1145/3377811.3380381.

[9] **Zishuo Ding, Jinfu Chen, and Weiyi Shang**. “Towards the Use of the Readily Available Tests from the Release Pipeline as Performance Tests: Are We There Yet?” In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1435–1446. ISBN: 9781450371216. DOI: 10.1145/3377811.3380351. URL: https://doi.org/10.1145/3377811.3380351.

[10] **Zhen Dong, Marcel Böhme, Lucia Cojocaru, and Abhik Roychoudhury**. “Time-Travel Testing of Android Apps”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 481–492. ISBN: 9781450371216. DOI: 10.1145/3377811.3380402. URL: https://doi.org/10.1145/3377811.3380402.

[11] **Thomas Durieux, João F. Ferreira, Rui Abreu, and Pedro Cruz**. “Empirical Review of Automated Analysis Tools on 47,587 Ethereum Smart Contracts”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 530–541. ISBN: 9781450371216. DOI: 10.1145/3377811.3380364. URL: https://doi.org/10.1145/3377811.3380364.

[12] **Tore Dybå, Dag I.K. Sjøberg, and Daniela S. Cruzes**. “What Works for Whom, Where, When, and Why? On the Role of Context in Empirical Software Engineering”. In: *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*. ESEM ’12. Lund, Sweden: Association for Computing Machinery, 2012, pp. 19–28. ISBN: 9781450310567. DOI: 10.1145/2372251.2372256. URL: https://doi.org/10.1145/2372251.2372256.

[13] **Andrew Forward and Timothy C. Lethbridge**. “A Taxonomy of Software Types to Facilitate Search and Evidence-Based Software Engineering”. In: *Proceedings of the 2008 Conference of the Center for Advanced Studies on Collaborative Research: Meeting of Minds*. CASCON ’08. Ontario, Canada: Association for Computing Machinery, 2008. ISBN: 9781450378826. DOI: 10.1145/1463788.1463807. URL: https://doi.org/10.1145/1463788.1463807.

[14] **Max Franz, Christian T. Lopes, Gerardo Huck, Yue Dong, Onur Sumer, and Gary D. Bader**. “Cytoscape.js: a graph theory library for visualisation and analysis”. In: *Bioinformatics* 32.2 (Sept. 2015), pp. 309–311. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btv557. eprint: https://academic.oup.com/bioinformatics/article-pdf/32/2/309/6689178/btv557.pdf. URL: https://doi.org/10.1093/bioinformatics/btv557.

[15] **Daniela Girardi, Nicole Novielli, Davide Fucci, and Filippo Lanubile**. “Recognizing Developers’ Emotions While Programming”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 666–677. ISBN: 9781450371216. DOI: 10.1145/3377811.3380374. URL: https://doi.org/10.1145/3377811.3380374.

[16] **Hui Guo and Cindy Rubio-González**. “Efficient Generation of Error-Inducing Floating-Point Inputs via Symbolic Execution”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1261–1272. ISBN: 9781450371216 DOI: 10.1145/3377811.3380359. URL: https://doi.org/10.1145/3377811.3380359.

[17] **Seongjoon Hong, Junhee Lee, Jeongsoo Lee, and Hakjoo Oh**. “SAVER: Scalable, Precise, and Safe Memory-Error Repair”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 271–283. ISBN: 9781450371216. DOI: 10.1145/3377811.3380323. URL: https://doi.org/10.1145/3377811.3380323.

[18] **Katherine Hough, Gebrehiwet Welearegai, Christian Hammer, and Jonathan Bell**. “Revealing Injection Vulnerabilities by Leveraging Existing Tests”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 284–296. ISBN: 9781450371216. DOI: 10.1145/3377811.3380326. URL: https://doi.org/10.1145/3377811.3380326.

[19] **Jingmei Hu, Jiwon Joung, Maia Jacobs, Krzysztof Z. Gajos, and Margo I. Seltzer**. “Improving Data Scientist Efficiency with Provenance”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1086–1097. ISBN: 9781450371216. DOI: 10.1145/3377811.3380366. URL: https://doi.org/10.1145/3377811.3380366.

[20] **Nargiz Humbatova, Gunel Jahangirova, Gabriele Bavota, Vincenzo Riccio, Andrea Stocco, and Paolo Tonella**. “Taxonomy of Real Faults in Deep Learning Systems”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1110–1121. ISBN: 9781450371216. DOI: 10.1145/3377811.3380395. URL: https://doi.org/10.1145/3377811.3380395.

[21] **Sungjae Hwang and Sukyoung Ryu**. “Gap between Theory and Practice: An Empirical Study of Security Patches in Solidity”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 542–553. ISBN: 9781450371216. DOI: 10.1145/3377811.3380424. URL: https://doi.org/10.1145/3377811.3380424.

[22] **Md Johirul Islam, Rangeet Pan, Giang Nguyen, and Hridesh Rajan**. “Repairing Deep Neural Networks: Fix Patterns and Challenges”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1135–1146. ISBN: 9781450371216. DOI: 10.1145/3377811.3380378. URL: https://doi.org/10.1145/3377811.3380378.

[23] **Gary Johns**. “The Essential Impact of Context on Organizational Behavior”. In: *Academy of Management Review* 31 (Apr. 2006). DOI: 10.5465/AMR.2006.20208687.

[24] **Brittany Johnson, Yuriy Brun, and Alexandra Meliou**. “Causal Testing: Understanding Defects’ Root Causes”. In: *Proceedings of the 42nd International Conference on Software Engineering (ICSE)* (ICSE). Seoul, Republic of Korea, June 2020, pp. 87–99. DOI: 10.1145/3377811.3380377.

[25] **Rafael-Michael Karampatsis, Hlib Babii, Romain Robbes, Charles Sutton, and Andrea Janes**. “Big Code != Big Vocabulary: Open-Vocabulary Models for Source Code”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1073–1085. ISBN: 9781450371216. DOI: 10.1145/3377811.3380342. URL: https://doi.org/10.1145/3377811.3380342.

[26] **Martin Kellogg, Manli Ran, Manu Sridharan, Martin Schäf, and Michael D. Ernst**. “Verifying Object Construction”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1447–1458. ISBN: 9781450371216. DOI: 10.1145/3377811.3380341. URL: https://doi.org/10.1145/3377811.3380341.

[27] **Lukas Kirschner, Ezekiel Soremekun, and Andreas Zeller**. “Debugging Inputs”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 75–86. ISBN: 9781450371216. DOI: 10.1145/3377811.3380329. URL: https://doi.org/10.1145/3377811.3380329.

[28] **Ryan Krueger, Yu Huang, Xinyu Liu, Tyler Santander, Westley Weimer, and Kevin Leach**. “Neurological Divide: An FMRI Study of Prose and Code Writing”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 678–690. ISBN: 9781450371216. DOI: 10.1145/3377811.3380348. URL: https://doi.org/10.1145/3377811.3380348.

[29] **Jason Lau, Aishwarya Sivaraman, Qian Zhang, Muhammad Ali Gulzar, Jason Cong, and Miryung Kim**. “HeteroRefactor: Refactoring for Heterogeneous Computing with FPGA”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 493–505. ISBN: 9781450371216. DOI: 10.1145/3377811.3380340. URL: https://doi.org/10.1145/3377811.3380340.

[30] **Ke Li, Zilin Xiang, Tao Chen, Shuo Wang, and Kay Chen Tan**. “Understanding the Automated Parameter Optimization on Transfer Learning for Cross-Project Defect Prediction: An Empirical Study”. In: ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 566–577. ISBN: 9781450371216. DOI: 10.1145/3377811.3380360. URL: https://doi.org/10.1145/3377811.3380360.

[31] **Michael Lienhardt, Ferruccio Damiani, Einar Broch Johnsen, and Jacopo Mauro**. “Lazy Product Discovery in Huge Configuration Spaces”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1509–1521. ISBN: 9781450371216. DOI: 10.1145/3377811.3380372. URL: https://doi.org/10.1145/3377811.3380372.

[32] **Kui Liu, Shangwen Wang, Anil Koyuncu, Kisub Kim, Tegawendé F. Bissyandé, Dongsun Kim, Peng Wu, Jacques Klein, Xiaoguang Mao, and Yves Le Traon**. “On the Efficiency of Test Suite Based Program Repair: A Systematic Assessment of 16 Automated Repair Systems for Java Programs”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 615–627. ISBN: 9781450371216. DOI: 10.1145/3377811.3380338. URL: https://doi.org/10.1145/3377811.3380338.

[33] **Peiming Liu, Gang Zhao, and Jeff Huang**. “Securing Unsafe Rust Programs with XRust”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 234–245. ISBN: 9781450371216. DOI: 10.1145/3377811.3380325. URL: https://doi.org/10.1145/3377811.3380325.

[34] **Valentin J. M. Manès, Soomin Kim, and Sang Kil Cha**. “Ankou: Guiding Grey-Box Fuzzing towards Combinatorial Difference”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1024–1036. ISBN: 9781450371216. DOI: 10.1145/3377811.3380421. URL: https://doi.org/10.1145/3377811.3380421.

[35] **George Mathew, Chris Parnin, and Kathryn T Stolee**. “SLACC: Simion-Based Language Agnostic Code Clones”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 210–221. ISBN: 9781450371216. DOI: 10.1145/3377811.3380407. URL: https://doi.org/10.1145/3377811.3380407.

[36] **Ehsan Mirsaeedi and Peter C. Rigby**. “Mitigating Turnover with Code Review Recommendation: Balancing Expertise, Workload, and Knowledge Distribution”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1183–1195. ISBN: 9781450371216. DOI: 10.1145/3377811.3380335. URL: https://doi.org/10.1145/3377811.3380335.

[37] **Jakob Nielsen and Thomas K. Landauer**. “A Mathematical Model of the Finding of Usability Problems”. In: *Proceedings of the INTERACT ’93 and CHI ’93 Conference on Human Factors in Computing Systems*. CHI ’93. Amsterdam, The Netherlands: Association for Computing Machinery, 1993, pp. 206–213. ISBN: 0897915755. DOI: 10.1145/169059.169166. URL: https://doi.org/10.1145/169059.169166.

[38] **Yannic Noller, Corina S. Păsăreanu, Marcel Böhme, Youcheng Sun, Hoang Lam Nguyen, and Lars Grunske**. “HyDiff: Hybrid Differential Software Analysis”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1273–1285. ISBN: 9781450371216. DOI: 10.1145/3377811.3380363. URL: https://doi.org/10.1145/3377811.3380363.

[39] **Andy Oram and Greg Wilson**. *Making Software: What Really Works, and Why We Believe It*. O’Reilly Media, Inc., 2010. ISBN: 0596808321.

[40] **Cassandra Overney, Jens Meinicke, Christian Kästner, and Bogdan Vasilescu**. “How to Not Get Rich: An Empirical Study of Donations in Open Source”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1209–1221. ISBN: 9781450371216. DOI: 10.1145/3377811.3380410. URL: https://doi.org/10.1145/3377811.3380410.

[41] **Profir-Petru Pârtachi, Santanu Kumar Dash, Christoph Treude, and Earl T. Barr**. “POSIT: Simultaneously Tagging Natural and Programming Languages”. In: ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1348–1358. ISBN: 9781450371216. DOI: 10.1145/3377811.3380440. URL: https://doi.org/10.1145/3377811.3380440.

[42] **Brandon Paulsen, Jingbo Wang, and Chao Wang**. “ReluDiff: Differential Verification of Deep Neural Networks”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 714–726. ISBN: 9781450371216. DOI: 10.1145/3377811.3380337. URL: https://doi.org/10.1145/3377811.3380337.

[43] **Harsha Perera, Waqar Hussain, Jon Whittle, Arif Nurwidyantoro, Davoud Mougouei, Rifat Ara Shams, and Gillian Oliver**. “A Study on the Prevalence of Human Values in Software Engineering Publications, 2015 – 2018”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 409–420. ISBN: 9781450371216. DOI: 10.1145/3377811.3380393. URL: https://doi.org/10.1145/3377811.3380393.

[44] **Kai Petersen and Claes Wohlin**. “Context in Industrial Software Engineering Research”. In: *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*. ESEM ’09. USA: IEEE Computer Society, 2009, pp. 401–404. ISBN: 9781424448425. DOI: 10.1109/ESEM.2009.5316010. URL: https://doi.org/10.1109/ESEM.2009.5316010.

[45] **Lutz Prechelt**. *Four presumed gaps in the software engineering research community's knowledge*. 2019. arXiv: 1911.09971 [cs.SE].

[46] **Akond Rahman, Effat Farhana, Chris Parnin, and Laurie Williams**. "Gang of Eight: A Defect Taxonomy for Infrastructure as Code Scripts". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 752–764. ISBN: 9781450371216. DOI: 10.1145/3377811.3380409. URL: https://doi.org/10.1145/3377811.3380409.

[47] **Sameer Reddy, Caroline Lemieux, Rohan Padhye, and Koushik Sen**. "Quickly Generating Diverse Valid Test Inputs with Reinforcement Learning". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1410–1421. ISBN: 9781450371216. DOI: 10.1145/3377811.3380399. URL: https://doi.org/10.1145/3377811.3380399.

[48] **Lin Shi, Mingzhe Xing, Mingyang Li, Yawen Wang, Shoubin Li, and Qing Wang**. "Detection of Hidden Feature Requests from Massive Chat Messages via Deep Siamese Network". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 641–653. ISBN: 9781450371216. DOI: 10.1145/3377811.3380356. URL: https://doi.org/10.1145/3377811.3380356.

[49] **Nischal Shrestha, Colton Botta, Titus Barik, and Chris Parnin**. "Here We Go Again: Why is It Difficult for Developers to Learn Another Programming Language?" In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 691–701. ISBN: 9781450371216. DOI: 10.1145/3377811.3380352. URL: https://doi.org/10.1145/3377811.3380352.

[50] **Thodoris Sotiropoulos, Dimitris Mitropoulos, and Diomidis Spinellis**. "Practical Fault Detection in Puppet Programs". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 26–37. ISBN: 9781450371216. DOI: 10.1145/3377811.3380384. URL: https://doi.org/10.1145/3377811.3380384.

[51] **Davide Spadini, Gül Çalikli, and Alberto Bacchelli**. "Primers or Reminders? The Effects of Existing Review Comments on Code Review". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1171–1182. ISBN: 9781450371216. DOI: 10.1145/3377811.3380385. URL: https://doi.org/10.1145/3377811.3380385.

[52] **Li Sui, Jens Dietrich, Amjed Tahir, and George Fourtounis**. "On the Recall of Static Call Graph Construction in Practice". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1049–1060. ISBN: 9781450371216

DOI: 10.1145/3377811.3380441. URL: https://doi.org/10.1145/3377811.3380441.

[53] **Guanhong Tao, Shiqing Ma, Yingqi Liu, Qiuling Xu, and X. Zhang**. "TRADER: Trace Divergence Analysis and Embedding Regulation for Debugging Recurrent Neural Networks". In: *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)* (2020), pp. 986–998.

[54] **Yuchi Tian, Ziyuan Zhong, Vicente Ordonez, Gail Kaiser, and Baishakhi Ray**. "Testing DNN Image Classifiers for Confusion amp; Bias Errors". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1122–1134. ISBN: 9781450371216. DOI: 10.1145/3377811.3380400. URL: https://doi.org/10.1145/3377811.3380400.

[55] **Rijnard van Tonder and Claire Le Goues**. "Tailoring Programs for Static Analysis via Program Transformation". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 824–834. ISBN: 9781450371216. DOI: 10.1145/3377811.3380343. URL: https://doi.org/10.1145/3377811.3380343.

[56] **Jue Wang, Yanyan Jiang, Chang Xu, Chun Cao, Xiaoxing Ma, and Jian Lu**. "ComboDroid: Generating High-Quality Test Inputs for Android Apps via Use Case Combinations". In: *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*. 2020, pp. 469–480.

[57] **Junjie Wang, Ye Yang, Song Wang, Yuanzhe Hu, Dandan Wang, and Qing Wang**. "Context-Aware in-Process Crowdworker Recommendation". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1535–1546. ISBN: 9781450371216. DOI: 10.1145/3377811.3380380. URL: https://doi.org/10.1145/3377811.3380380.

[58] **Ying Wang, Ming Wen, Yepang Liu, Yibo Wang, Zhenming Li, Chao Wang, Hai Yu, Shing-Chi Cheung, Chang Xu, and Zhiliang Zhu**. "Watchman: Monitoring Dependency Conflicts for Python Library Ecosystem". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 125–135. ISBN: 9781450371216. DOI: 10.1145/3377811.3380426. URL: https://doi.org/10.1145/3377811.3380426.

[59] **Cheng Wen, Haijun Wang, Yuekang Li, Shengchao Qin, Yang Liu, Zhiwu Xu, Hongxu Chen, Xiaofei Xie, Geguang Pu, and Ting Liu**. "MemLock: Memory Usage Guided Fuzzing". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 765–777. ISBN: 9781450371216. DOI: 10.1145/3377811.3380396. URL: https://doi.org/10.1145/3377811.3380396.

[60] **Wikipedia**. *Association for Computing Machinery — Wikipedia, The Free Encyclopedia*. http://de.wikipedia.org/w/index.php?title=Association%20for%20Computing%20Machinery&oldid=203336551. [Online; accessed 23-August-2021]. 2021.

[61] **Wikipedia**. *Marktsegmentierung — Wikipedia, The Free Encyclopedia*. http://de.wikipedia.org/w/index.php?title=Marktsegmentierung&oldid=201060729. [Online; accessed 23-August-2021]. 2021.

[62] **Jiwei Yan, Hao Liu, Linjie Pan, Jun Yan, Jian Zhang, and Bin Liang**. "Multiple-Entry Testing of Android Applications by Constructing Activity Launching Contexts". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 457–468. ISBN: 9781450371216. DOI: 10.1145/3377811.3380347. URL: https://doi.org/10.1145/3377811.3380347.

[63] **Rahulkrishna Yandrapally, Andrea Stocco, and Ali Mesbah**. "Near-Duplicate Detection in Web App Model Inference". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 186–197. ISBN: 9781450371216. DOI: 10.1145/3377811.3380416. URL: https://doi.org/10.1145/3377811.3380416.

[64] **Junwen Yang, Utsav Sethi, Cong Yan, Alvin Cheung, and Shan Lu**. "Managing Data Constraints in Database-Backed Web Applications". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1098–1109. ISBN: 9781450371216. DOI: 10.1145/3377811.3380375. URL: https://doi.org/10.1145/3377811.3380375.

[65] **Hengbiao Yu, Zhenbang Chen, Xianjin Fu, Ji Wang, Zhendong Su, Jun Sun, Chun Huang, and Wei Dong**. "Symbolic Verification of Message Passing Interface Programs". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1248–1260. ISBN: 9781450371216. DOI: 10.1145/3377811.3380419. URL: https://doi.org/10.1145/3377811.3380419.

[66] **Juan Zhai, Xiangzhe Xu, Yu Shi, Guanhong Tao, Minxue Pan, Shiqing Ma, Lei Xu, Weifeng Zhang, Lin Tan, and Xiangyu Zhang**. "CPC: Automatically Classifying and Propagating Natural Language Comments via Program Analysis". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1359–1371. ISBN: 9781450371216. DOI: 10.1145/3377811.3380427. URL: https://doi.org/10.1145/3377811.3380427.

[67] **Peixin Zhang, Jingyi Wang, Jun Sun, Guoliang Dong, Xinyu Wang, Xingen Wang, Jin Song Dong, and Ting Dai**. "White-Box Fairness Testing through Adversarial Sampling". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Associa-

tion for Computing Machinery, 2020, pp. 949–960. ISBN: 9781450371216. DOI: 10.1145/3377811.3380331. URL: https://doi.org/10.1145/3377811.3380331.

[68] **Ru Zhang, Wencong Xiao, Hongyu Zhang, Yu Liu, Haoxiang Lin, and Mao Yang**. “An Empirical Study on Program Failures of Deep Learning Jobs”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1159–1170. ISBN: 9781450371216. DOI: 10.1145/3377811.3380362. URL: https://doi.org/10.1145/3377811.3380362.

[69] **Xueling Zhang, Xiaoyin Wang, Rocky Slavin, Travis Breaux, and Jianwei Niu**. “How Does Misconfiguration of Analytic Services Compromise Mobile Privacy?” In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 1572–1583. ISBN: 9781450371216. DOI: 10.1145/3377811.3380401. URL: https://doi.org/10.1145/3377811.3380401.

[70] **Changwei Zou and Jingling Xue**. “Burn after Reading: A Shadow Stack with Microsecond-Level Runtime Rerandomization for Protecting Return Addresses”. In: ICSE ’20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 258–270. ISBN: 9781450371216. DOI: 10.1145/3377811.3380439. URL: https://doi.org/10.1145/3377811.3380439.