OTTO-FRIEDRICH-UNIVERSITÄT
BAMBERG

Qualitative Analyse von Stakeholdern in einem Unternehmen zu Sichtweisen auf Privatsphäre bei Softwareentwicklern bzw. -administratoren

VON

Samet Murat Akcabay

Lehrstuhl für Privatsphäre und Sicherheit in Informationssystemen

2. August 2020

Zusammenfassung

Diese qualitative Analyse beschäftigt sich mit den Sichtweisen auf Privatsphäre und Sicherheit von Stakeholdern in einem Softwareentwicklungs-Unternehmen. Mit der stetig wachsenden Digitalisierung spielt der Datenschutz, und somit die Privatsphäre und Sicherheit, eine immer größere Rolle bei Nutzern des Internets. Aus praktischer Sicht sind die Sichtweisen auf diese unterschiedlich, weswegen bei der Recherche darauf Wert gelegt wurde, möglichst viele Experten aus unterschiedlichen Berufsgruppen (z.B. Werkstudenten, Festangestellte, Team-Leiter o.Ä.) einzubeziehen und deren Aussagen auszuwerten. Nach erfolgreicher Auswertung der Expertenbefragungen hat sich ergeben, dass die Berufsgruppe, damit verbunden die Arbeitsweise und der -umfang, [...] keinen Einfluss auf die Einstellung gegenüber personenbezogenen Daten und die Verarbeitung dieser in DevOps-Tools haben.

Inhaltsverzeichnis

1 Die Relevanz der Privatsphäre im 21. Jahrhundert - eine Einleitung

„We believe privacy is a fundamental human right“ [App20] - mit dieser Aussage betritt Craig Federighi, der Vizepräsident der Softwareentwicklung von Apple, einem erfolgreichen Multimilliarden-Unternehmen der Software- und Hardwareentwicklung aus den Vereinigten Staaten, zur Vorstellung der neuen Privatsphäre-Richtliniendes Unternehmens zur sogenannten, firmeneigenen „Worldwide Developer Conference“ die Bühne.
Und dieser Glaube ist nicht unbegründet: Mit der zunehmenden Digitalisierung in den letzten Jahrzehnten werden Menschen, und Internetnutzer im Spezifischen, häufiger vor dieses Dilemma gestellt. Entwicklungen wie dem Internet der Dinge (IoT), Industrie 4.0, „Smart Home“, Wearables (diverse Sensoren an der Kleidung selbst, Smartwatches, Fitnesstracker) uvm. ermöglichen neue (und zum Teil unerforschte) Angriffsmöglichkeiten für Cyberkriminalität [Bun19]. Dies zeichnet sich vor allem durch den Anstieg der Höhe der Fallzahlen und der gleichzeitigen Abnahme der Entdeckungsquote der Malware aus: Im ersten Halbjahr 2018 wurden im Durchschnitt 13.000 Malware-Samples am Tag neu entdeckt [18b], während diese Zahl im zweiten Halbjahr 2017 noch bei knapp 23.000 lag [18a].
Ähnlich sieht es hier bei dem Anstieg neuer Schadprogramme aus: Der G Data Blog, ein deutsches Softwareunternehmen, welches mehrfach für seine Sicherheitslösungen ausgezeichnet wurde [20b], beschreibt dabei, dass die Zahl der neuen Schadprogrammtypen seit 2007 einen Anstieg um das knapp 63-fache vermerken konnte [18a]. Es ist anzunehmen, dass dies sicherlich auch damit zusammenhängt, dass das Internet in den letzten beiden Jahrzehnten einen regelrechten Ausbruch in der Nutzung erlebt hat. Während die Computer- und Internetnutzung 2012, einem Jahr, in welchem im Durchschnitt bereits knapp 46% der Generation Plus50 Internetnutzer waren [16], noch für private Haushalte und Personen ab 10 Jahren bei 78% lag [19], ist die Internetnutzung 2019 bis auf 88% angestiegen [19].

Nun könnte man anhand der steigenden Tendenz davon ausgehen, dass

das Internet sich etabliert und die Nutzer durch den langjährigen Umgang und das gleichzeitige Sammeln von Erfahrung ein allgemeines Wissen über die Risiken und Gefahren im Bezug auf die eigene Sicherheit und Privatsphäre aufgebaut haben. Bei einer Umfrage durch Bitkom Research 2018 haben 89% in der Altersgruppe ab 14 Jahren angegeben, sich in der Regel mit den Privatsphäre-Einstellungen auseinandergesetzt zu haben [Bit18]. Auf der Kehrseite sieht man jedoch, wie beispielsweise diverse Bonusprogramme der DeutschlandCard und Payback GmbH, welche häufig in der Kritik von Verbraucherschützern stehen, da Nutzerdaten zu den Einkäufen und dem Einkaufsverhalten gesammelt werden [HD], immer größer und verbreiteter werden: Alleine zum Vorjahr konnte die Payback GmbH im Jahre 2018 einen Anstieg des Jahresumsatzes auf 330,49 Millionen Euro in Deutschland und Österreich verzeichnen, während 2012 der Umsatz noch bei 148,57 Millionen Euro lag [Pay19].
Dieses Phänomen, auf der einen Seite seine Privatsphäre zu wahren und schützen wollen, auf der anderen Seite jedoch seine persönlichen Daten freiwillig auf Sozialen Netzwerken, beim Einkaufen oder durch die eigene Art und Weise, ein internetfähiges Gerät zu bedienen, zu teilen, nennt man das „Privacy Paradox“ (dt. *Privatsphäre-Paradoxon*) [Bar06] und wird im weiteren Verlauf eine Rolle spielen.

2 Theoretischer Hintergrund

Um in den folgenden Kapiteln ein grundlegendes Verständnis der Begrifflichkeiten, Zusammenhänge und Konzepte gewährleisten zu können, ist es notwendig, diese vorab zu definieren und einzugrenzen. Da sich diese Arbeit mit der Privatsphäre von Stakeholdern in einem Softwareentwicklungsunternehmen beschäftigt, ist es zunächst erforderlich, die Grenzen der Privatsphäre klar zu definieren und anhand dieser, eine Messung zu schaffen. Des weiteren müssen die Stakeholder, welche in dieser Analyse betrachtet werden, definiert, identifiziert und im weiteren Verlauf in Gruppen gegliedert werden. Im Anschluss werden DevOps definiert und anhand gewählter Beispiele, eine Analyse dieser aufgebaut.

2.1 Die Privatsphäre - eine Eingrenzung

Der Begriff Privatsphäre ist im Wortschatz der deutschen Sprache tief verankert und findet im Alltag in vielerlei Hinsicht Gebrauch: Menschen decken Notebook- und Smartphonekameras ab, protestieren gegen Überwachungskameras im öffentlichen Raum [Sta13] oder möchten nicht, dass ihre persönlichen und privaten Daten auf sozialen Netzwerken ohne ihre Zustimmung bzw. ohne ihre Kenntnis verbreitet werden [Pic18] - doch was genau bedeutet Privatsphäre?
Die Privatsphäre lässt sich definieren als einen eingegrenzten, nichtöffentlichen Raum, in welchem ein Individuum bzw. Individuen nach eigenem Belieben, ohne äußere Einflüsse oder die Beobachtung durch Unbeteiligte, zur freien Entfaltung der eigenen Person, handeln kann [Pet20]. Möchte man diese Definition nun auf die Softwareentwicklung widerspiegeln, so müssen die möglichen, vorherrschenden Komponenten von persönlichen Daten näher betrachtet werden: In Unternehmen, welche sich mit der Softwareentwicklung befassen, müssen Daten vorhanden sein, welche Individuen individuell und bestimmen können (Grund liefern!) - dies kann in Form von IDs und individuell gewählten Nutzernamen vorzufinden sein, aber auf der Kehrseite auch mit Klarnamen, E-Mail Adressen sowie eindeutig identifizierbaren IDs.

2.2 Stakeholder

Der Begriff „Stakeholder“ lässt sich neben der Informatik auch in der Betriebswirtschaftslehre vorfinden - das Bindeglied der beiden Bereiche in diesem Aspekt stellen Unternehmen im Allgemeinen selbst dar. Da diverse Institutionen, Personen und Personengruppen Erwartungen an Unternehmen haben und eigene Interessen vertreten, stellen Stakeholder jene „[...] [dar], die von den Aktivitäten eines Unternehmens direkt oder indirekt betroffen sind oder [...] ein Interesse an diesen [...] haben“ [Fle16]. Als Beispiele können diese in Form von Kunden und Lieferanten bis zu eigenen Mitarbeitern und Eigentümern (von Unternehmensanteilen oder vom Unternehmen selbst) auftreten [Fle16].
Neben den Stakeholdern selbst existiert der sogenannte „Stakeholder-Ansatz“: Dieser besagt die Betrachtung einer Unternehmung als Organisation unter Zusammenschluss verschiedener Stakeholder [Sey18]. Dabei stellt die Unternehmensleitung die Vermittlung zwischen den verschiedenen Gruppen dar [Sey18].
Betrachtet man diesen Begriff nun aus der informatischen Sicht, so kann man aus der Definition herleiten, dass Stakeholder hier in der Regel Erwartungen und Interessen an einem einem Programm bzw. einer Applikation oder einem Computersystem, statt einem Unternehmen verfolgen. dieser Analyse ist der Bezug fortlaufend auf diese Stakeholder gelegt.

2.3 DevOps

„Die Zufriedenheit unserer Kunden liegt uns am Herzen“ - durch dieses Motto strahlen Unternehmen ihre Kundenorientierung aus und möchten in der Regel zusätzlich durch Feedback, Kulanz, einem überzeugenden Produkt oder einem allgemein positiv zurückgebliebenem Eindruck beim Kunden überzeugen. In Softwareentwicklungsunternehmen vereint DevOps die benötigten Technologien, die Prozesse und die Menschen miteinander, um für diese Kunden andauernd hochwertige Produkte anbieten zu können [20c]. Der Begriff setzt sich zusammen aus „Development“ (dt. *Entwicklung*) von Software sowie „Operations“ (dt. *Vorgänge*), welche damit zusammenhängen [20c] und repräsentiert dabei die „Zusammenarbeit von Entwicklung und Betrieb“ [Has15] dar. Dies erfolgt durch die Einführung von DevOps-Methoden und -Tools, sodass den zuvor getrennten Bereichen, und zusätzlichen wie beispielsweise der Qualitätssicherung, der Sicherheit etc., die Möglichkeit zur Koordination und Zusammenarbeit geboten wird [20c].
Im Folgenden werden beispielhaft repräsentative DevOps-Methoden und -Tools definiert und aufgelistet.

2.3.1 Continuous Integration

In der Softwareprogrammierung ist es mit zunehmender Komplexität und Umfang des Programms üblich, mehrere einzelne Module in dieses einzubinden, weswegen sich die gängige Vorgehensweise in der Softwareentwicklung insofern gestaltet, dass das Gesamtprojekt, welches aus

diesen einzelnen Modulen besteht, durch mehrere Teams parallel vorangetrieben [PA18]. Statt auf eine täglich erfolgende Kompilierung auf einem extra angelegten „Build"-Server warten zu müssen, kommt heutzutage in modernen Unternehmen die sogenannte „Continuous Integration" (dt. *Kontinuierliche Integration*, kurz: CI) zum Einsatz [PA18]: Diese bezeichnet „eine Technik der agilen Softwareentwicklung [...], [welche die] in kleinen Schritten vorgenommenen Änderungen [...] regelmäßig zusammenführt" [Dir18], also eine kontinuierliche Integration von Softwarekomponenten wie z.B. eines Guided User Interfaces (dt. *grafische Benutzeroberfläche*, kurz: GUI), eines Application Programming Interfaces (dt. *Programmierschnittstelle*) etc., in das gemeinsame Projekt bzw. den gemeinsamen Code [PA18]. Dies erfolgt in der Regel unter Einsatz von einem sogenannten „Trunk", also der „Stamm", welches das Hauptverzeichnis der Entwicklung des Projekts darstellt und „Branches", die als „Äste" bzw. Verzweigungen angesehen werden können und die untergeordneten Verzeichnisse der Hauptentwicklung repräsentieren [Pet15].
In der Arbeit im Team kann der Code dann für dasselbe Projekt also nicht erst wie zuvor nach dem Fertigstellen der anderen Teilbereiche mit in den Gesamtcode des Projekts aufgenommen werden, sondern nach eigenem Bedarf, sogar mehrmals am Tag [Dir18]. Dadurch kann der Code für die beteiligten Programmierer schon viel früher offengelegt werden, sodass Tests und damit verbunden, Fehlerbehebungen früher und effizienter durch Einsparung von Zeit und Ressourcen, durchgeführt werden können - wobei trotzdem zunächst eine Umorientierung erforderlich ist, da ein bereits gängiger Prozess neu aufgelegt und zusätzliche Prozesse sowie Server neu eingebunden werden müssen [Dir18; PA18]. Dieser gesamte Prozess funktioniert durch das Einspeisen der zusammengefügten Änderungen des Codes in ein Repository, womit im Anschluss eine Versionskontrolle durch die Übernahme von einer aktuellen Version des Codes durch einen automatisierten Build-Prozess [PA18].

Abbildung 2.1: Die CI-/CD-Pipeline.

2.3.2 Continuous Delivery

Der Begriff „Continuous Delivery" (dt. *Kontinuierliche Auslieferung*, kurz: CD) geht einen Schritt weiter: Eine Sammlung, welche „kurze Entwicklungszyklen und [eine] schnelle Auslieferung von Software-Updates [durch Techniken, Prozesse und Werkzeuge]" [IA17] ermöglicht, beschreibt man als Continuous Delivery. Anstatt konventionell darauf zu setzen, Software erst zu programmieren und im Anschluss einem Test und einer Evaluation zu unterziehen, bevor diese an Kunden ausgeliefert wird, kann durch Continuous Delivery eine vollfunktionsfähige Version der Software in jedem Entwicklungsstand nach Bedarf vom Kunden bezogen werden - in der Regel ohne Fehler zu enthalten [IA17].

2.3.3 Continuous Deployment

Das Continuous Deployment (kurz: CD) stellt den Abschluss der CI-/CD-Pipeline (s. Abbildung 2.1) dar und erweitert den Continuous Delivery, welche die zur Auslieferung fertigen Builds automatisch freigibt, um die automatische Freigabe einer Applikation in der Phase der Produktion

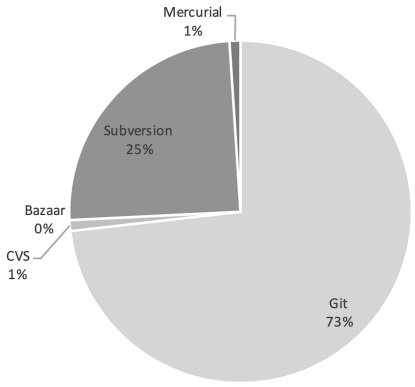

Abbildung 2.2: Versionsverwaltungs-Tools im Ranking (nach Anzahl der Repositories) - Stand: 2020.

[20d]. Dies bedeutet in der Praxis betrachtet, dass nach Fertigstellen der Codeänderungen und nach erfolgreichem Bestehen der definierten Tests, die Applikation automatisch (bspw. an die Kunden zum Einholen von Feedback) ausgeliefert werden kann [20d].

2.3.4 Versionsverwaltung

Durch die Zusammenarbeit im Team an einem gemeinsamen Projekt werden Softwareentwickler häufig vor das Problem der Versionsverwaltung gestellt: Wie können mehrere Personen an einem gemeinsamen Projekt und z.T. sogar am selben Programm gleichzeitig arbeiten? Wie können verschiedene Versionen eines Programms einheitlich und übersichtlich verwaltet werden? Und wie verfährt man möglichst effizient mit Code, welcher von unterschiedlichen Autoren geschrieben wurde und fügt diese zusammen? Zur Lösung dieser Konflikte ist die Versionsverwaltung heutzutage in Softwareentwicklungsunternehmen essentiell. Diese „ist ein System, welches die Änderungen an einer oder einer Reihe von Dateien über die Zeit hinweg protokolliert, sodass man später auf eine bestimmte Version zurückgreifen kann." [CS20]. Laut der Webseite Open Hub, welche zur Katalogisierung von Open-Source-Software verwendet wird und wie hier, den Anteil von Versionsverwaltungs-Tools anhand der registrierten Daten indexiert, stellen Git (71%) und Subversion (24%) die Spitze dar (s. Abbildung 2.2) [Inc20], weswegen diese in den folgenden Unterkapiteln näher betrachtet werden.

2.3.4.1 Git

Git ist eine freie Software zur Versionsverwaltung [20a]. Die grundlegende Funktion unterscheidet hierbei zwischen der lokalen, zentralen und verteilten Versionsverwaltung: Die lokale Versionsverwaltung befasst sich dabei mit der „[Verwaltung] aller Änderungen relevanter Dateien in einer Datenbank" [CS20], wohingegen die zentrale mit dem Problem der Kooperation mit anderen Entwicklern an gemeinsamer Software [CS20]. Der letzte Punkt der verteilten Verwaltung behandelt das vollständige Kopieren eines sogenannten „Repositories" (dt. *Quelle bzw. Lager von Software*) von einem Server anstatt lediglich den letzten Stand zu beziehen [CS20]. Unter Git wird das Hauptverzeichnis als „master" bezeichnet, wohingegen die Versionierungen hiervon als „Branches" (vgl. Abbildung 2.3), wie unter SVN (s. Kapitel 2.3.4.2), betitelt werden.

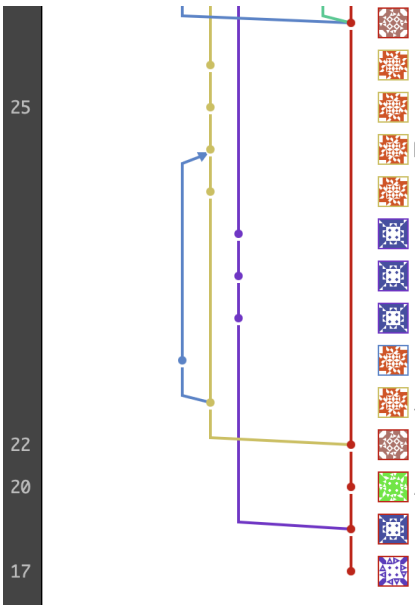

Abbildung 2.3: Beispielhafte Historie einer Versionsverwaltung (hier: Git; Verschiedene Profilbilder repräsentieren einzelne Commit-Autoren, Abzweigungen repräsentieren neue Branches))

2.3.4.2 Subversion

Apache Subversion (kurz: SVN) ist ebenfalls eine freie Software zur Versionsverwaltung von Dateien sowie Verzeichnissen, wodurch alte Versionen von Daten und eine Historie der verschiedenen Änderungen und Versionen an diesen betrachtet werden können [CFP04]. Unter Subversion wird das Verzeichnis der „Hauptlinie" als „Trunk" definiert und seine Subverzeichnisse, wie auch schon unter Git (s. Kapitel 2.3.4.1), als „Branches".

3 Forschungsmethode

Aufgrund der Natur der offenen Befragung wurde hierbei auf Leitfadeninterviews zurückgegriffen. Dadurch wurde gewährleistet, dass das Forschungsthema im Fokus der Befragung bleibt und Thematiken, welche von den Interviewpartnern angesprochen wurden, näher spezifiziert und erläutert werden konnten. Hier wurden insgesamt sieben Interviews mit Personen durchgeführt, welche aus zum Teil verschiedenen Bereichen und Positionen in Softwareentwicklungsunternehmen stammen, aber dennoch mit DevOps und der Softwareentwicklung in Berührung stehen.

Besonders wichtig hierbei war es, soweit möglich, unterschiedliche Experten aus unterschiedlichen Bereichen und Positionen eines Unternehmens zu finden. Dabei waren alle Teilnehmer zum Zeitpunkt der Befragung über 23 Jahre alt. Fünf der sieben Interviews wurden hierbei bereits vor Beginn dieser Analyse vom Lehrstuhl für Privatsphäre und Sicherheit in Informationssystemen (kurz: PSI) durchgeführt und transkribiert, weswegen diese weiterverwendet und codiert werden konnten. Zwei der insgesamt sieben Interviews wurden während der Analyse durchgeführt, wovon eine persönlich und durch Audio-Aufzeichnung unterstützt und die Letzte durch ein zu ausfüllendes Online-Formular stattgefunden hat, da ein persönliches Gespräch aufgrund der COVID-19 Pandemie nicht möglich war. Einen Einfluss auf die Ergebnisse sollten diese Unterschiede in der Demographie und in der Durchführung jedoch nicht haben.
Der Leitfaden wurde vorab vom PSI-Lehrstuhl zur Verfügung gestellt und für diese Analyse weiterverwendet. Dieser ist im Anhang einsehbar (s. Anlage [...]).

3.1 Vorbereitung

Bevor die Experteninterviews durchgeführt wurden, mussten Vorarbeiten geleistet werden. Zunächst war es wichtig, eine Datenschutzerklärung zu erstellen, welche die zu interviewenden Personen über den Umgang mit ihren persönlichen Daten aufgeklärt hat. Dadurch konnte zunächst gewährleistet werden, dass es sich bei dieser Forschung um ein seriöses Vorgehen handelt. Des weiteren konnte so den Interviewpartnern eine rechtliche Grundlage über ihren Datenschutz und der persönlichen Daten zugesichert werden. Außerdem wurde angegeben, mit der Teilnahme an der Befragung der Datenschutzerklärung automatisch zuzustimmen. Parallel dazu wurde eine Ausschreibung erstellt, welche das Interview

beworben hat und in diversen sozialen Netzwerken (Twitter, LinkedIn, XING etc.) geteilt werden konnte. Hier wurde angegeben, Personen im Rahmen dieser Analyse zu suchen, welche bereits in einem Softwareentwicklungsunternehmen tätig waren oder tätig sind, Erfahrungen mit der Arbeit im Team und mit DevOps besitzen und grobe Kenntnisse über interne Abläufe (in Bezug auf Verwaltung des Unternehmens, Abläufe im Team, aber auch mit DevOps) besitzen.

Zu Personen, welche auf die Ausschreibung reagiert haben, wurde über E-Mail Kontakt aufgenommen und ein Termin zum gemeinsamen Treffen vereinbart. Die Befragung wurde in einem Seminarsaal durchgeführt und währenddessen mit einem Notebook aufgezeichnet. Da eine persönliche Befragung nach Ausbruch der COVID-19 Pandemie nicht mehr möglich war, wurde mithilfe von „Microsoft Forms“ ein Online-Formular mit den Leitfragen erstellt und als Direktlink per E-Mail an die Interessenten weitergeleitet. Hierbei war es besonders wichtig, die Leitfragen so zu wählen, dass ein grundlegendes Verständnis, der Sinn und die Zusammenhänge dieser gewährleistet waren, da eine nachträgliche Erläuterung und Spezifizierung aufgrund der gegebenen Umstände nicht mehr möglich waren.

3.2 Durchführung und Transkription der Experteninterviews

Zu Beginn des Interviews wurde der zu Interviewende begrüßt und das Einverständnis zur Aufnahme des Gesprächs eingeholt. Danach wurde die Aufnahme gestartet und die Bestätigung zur Kenntnisnahme der Datenschutzerklärung sowie der Vermerk, diesem jederzeit widersprechen zu können erläutert. Da in der Stellenausschreibung und im darauffolgenden in Kontakt treten kommuniziert wurde, DevOps und diverse andere Arbeitsabläufe zu kennen, wurde darauf verzichtet, im Vorfeld Erläuterungen und Definitionen zu Begriffen wie DevOps und den Tools, Stakeholder oder Vergleichbares zu klären. Dieses Vorgehen wurde durch den durchweg flüssigen Dialog zusätzlich bestätigt. Im Interview selbst wurde sich strikt an den Leitfaden gehalten: Sofern eine Abweichung der Antworten und somit eine Richtungsänderung stattgefunden hat, wurde, passend zum vorgegeben Leitfaden, eine „Alternativroute“ der Fragestellungen eingeschlagen.

Zur Abgrenzung der verschiedenen Stakeholder war es zu Anfang wichtig, welche Position im Unternehmen repräsentiert wurde und ob diese bereits oder gegenwärtig mit der Administration von Servern beauftragt waren bzw. sind, da auf diese Art gezielt Fragen in Bezug auf diese Merkmale gestellt werden konnten (s. Tabelle 3.1). In den Befragungen stand im Vordergrund, die Sichtweise auf die Verarbeitung von eigenen und fremden persönlichen Daten einzuholen. Falls verweigert wurde, eigene persönliche Daten (vor allem in DevOps) preiszugeben, wurden Alternativen angeregt. Zudem sollten die Experten dazu angeregt werden, ihre eigene Meinung und Denkweise zu präsentieren, wie der Umgang mit persönlichen Daten in anderen Unternehmen bzw. unabhängig von

Tabelle 3.1: Position und Erfahrung der Stakeholder im Unternehmen.

	Beobachtete Ergebnisse	
Rolle	**Administration von Servern**	**Nutzung von Log-Dateien**
Softwareentwickler	25% Ja, 75% Nein	Ja (Server- und Applikationslogs)
Softwareentwickler und Berater	Nein	Keine Angabe
Lead-Developer	Ja	Ja (Fehler- und Zugangslogs)
Systemadministrator und DevOps	Ja	Ja (Alle Logs)

der eigenen Person stattfindet. Bestärkt wurde dieser Aspekt durch die spezifische Frage nach Nutzung von Log-Dateien: Die Intention hierbei war dabei, die Experten indirekt zu fragen, ob diese zunächst Einsicht in persönliche Daten von Kollegen durch Logs haben und ob sie diese Möglichkeit auch tatsächlich aktiv nutzen.

Die Befragungen fanden im Zeitraum zwischen März und Juli 2020 statt, wobei als spätmöglichster Tag zur Durchführung einer Befragung als der 15. Juli 2020 festgelegt wurde. Diese dauerten im Durchschnitt zehn Minuten. Im Anschluss erfolgte eine Transkription der Interviews. Dieses Prinzip wird auch die „fallbezogene Auswertung“ [DB14] genannt - hierbei wird das qualitative Datenmaterial sequenziell, also von Anfang bis Ende, nach Fällen ausgewertet und im Anschluss kodiert [DB14].

Hierzu wurden die Aufnahmen wiederholt und die Fragestellungen mit den entsprechenden Antworten mittels einer Markdown-Datei festgehalten. Dieses Vorgehen wurde bei der Befragung in Form einer Umfrage redundant, da dieses als PDF-Dokument exportiert werden konnte. Zur fristgerechten Fertigstellung der Analyse und der gleichzeitigen Behinderung durch die COVID-19 Pandemie wurde sich nach Absprache mit der Leitung des PSI-Lehrstuhls darauf geeinigt, die Anzahl der Befragungen auf sieben zu limitieren.

3.3 Kodierung der Ergebnisse

Zur Kodierung der Transkriptionen aus den Befragungen wurden die Methoden nach Döring und Bortz gewählt: Den Textstellen (entspricht einem Teil des qualitativen Datenmaterials) aus den Ergebnissen wurden hierbei Codes zugewiesen, welcher eine Zusammenfassung bzw. eine Erklärung dieser darstellt [DB14] (vgl. Tabelle 3.2) - dieses Vorgehen steht die Vorgehensweise der deskriptiven Kodierung dar, mit dessen Hilfe ähnliche Aussagen zusammengefasst werden können. Zum Schluss werden diese Codes noch ihren entsprechen, zugehörigen Kategorien, welche im weiteren Verlauf dieser Analyse die jeweiligen Sichtweisen der einzelnen Stakeholder auf die Privatsphäre darstellt, zugeteilt, um eine allgemeine Abgrenzung zwischen diesen gewährleisten zu können.

Tabelle 3.2: Repräsentatives Codierungs- und Kategorisierungsbeispiel der erhobenen Daten (in Anlehnung an Döring und Bortz [DB14])

		Erhobe
Textstelle	**Code**	
„Mich interessiert nicht, wer den Build zerstört hat."	Desinteresse	
„Mir ist es wichtig, dass mein Team nicht nur im Chat rumhängt."	Nutzeraktivitäten	Aktivitäten des
„Ich nutze Slack nur wegen seiner Emojis."	Desinteresse	

4 Ergebnisse

Anhand der Expertenbefragungen konnten insgesamt zwölf verschiedene Kategorien zu den jeweiligen Sichtweisen der einzelnen Stakeholder herausausgearbeitet werden. Unterschieden wird hierbei zwischen Kategorien mit einstimmigen in Kapitel 4.1 und abweichenden Ergebnissen in Kapitel 4.2, wobei die genaue Bedeutung dieser Eigenschaften in den entsprechenden Kapiteln näher erläutert werden. Die Alternativen zu den einzelnen Aspekten, welche von den einzelnen Stakeholdern vorgeschlagen wurden, werden im Kapitel 4.3 aufgelistet. Als Letztes werden im Kapitel **??** die Ergebnisse tabellarisch präsentiert.

4.1 Kategorien mit einstimmigen Ergebnissen

In diesem Kapitel werden die einzelnen Kategorien anhand der Einstimmigkeit der einzelnen Kategorien aufgelistet. Einstimmig bedeutet in diesem Fall, dass bei den einzelnen Kategorien eine Abweichung von maximal drei von sieben Befragten (<50%) stattgefunden hat - also anhand der absoluten Häufigkeit eine eindeutige Aussage getroffen werden kann. Hierbei wird zwischen der Präferenz der Privatsphäre im Allgemeinen oder der Benutzerfreundlichkeit bzw. Einfachheit unterschieden: Kategorien, in welchen Experten überwiegend (>50% der Experten sind der selben Ansicht) angegeben haben, die Privatsphäre sei ihnen wichtiger, werden dem Kapitel 4.1.2 zugeordnet, wohingegen die Kategorien, in welchen die Privatsphäre aufgrund der Benutzerfreundlichkeit und Einfachheit vernachlässigt werden konnte, dem Kapitel 4.1.1 zugerechnet werden.

4.1.1 Kategorien mit einer Präferenz für die Benutzerfreundlichkeit und Einfachheit

Verarbeitete Daten zur Unterstützung der Softwareentwicklung in DevOps-Tools:
Zu Beginn der Interviews haben alle Befragten angegeben, mit persönlichen Daten in Berührung zu kommen und diese auch zu verarbeiten. Dabei haben alle vier Softwareentwickler angegeben, die Nutzeraktivitäten und den Programmcode von Kollegen einsehen und verarbeiten zu können. Diese genannten Daten beinhalten Benutzernamen, Zeitstempel (in Git Commits, Builds und Deployments, z.T.) Dies erfolge zur Nachvollziehbarkeit der Builds und Deployments, des Programmcodes selbst,

z.B. durch Git Commits und zur allgemeinen Fehlerbehebung bei fehlgeschlagenen Builds bzw. fehlerhaftem Programmcode. Zwei Befragte haben zudem angegeben, Zugriff auf sensible Kundendaten in Form von Klarnamen, Anschriften, E-Mail Adressen uvm. zu besitzen: Dies ist bei jenen der Fall, die häufig mit externen Ansprechpartnern und Unternehmen in Kontakt treten und mit diesen zusammenarbeiten müssen.

Bevorzugter Detailgrad von Zeitstempeln in DevOps-Tools:
Dieser Aspekt nimmt genaueren Bezug auf Zeitstempel in DevOps-Tools und befasst sich mit der Auflösung (z.B. Detailgrad, Lebensdauer und relevante Faktoren) dieser. Hier wurden die Experten befragt, in welcher Auflösung diese konkret den Zeitstempel im Arbeitsalltag benötigen. Bis auf eine Ausnahme, haben alle Befragten angegeben, einen konkreten Zeitstempel mit der Anfangs- und Endzeit (bzw. alternativ mit der Anfangszeit und der Dauer) zu benötigen und auf diesen nicht verzichten zu können. Dies sei vor allem in der Fehlerbehebung (z.B. von fehlerhaften Commits) elementar, denn nur durch eine genaue Zeitangabe könne man den Fehler in Logs, im Code selbst („Wie lange hat es gedauert, bis der Fehler aufgetreten ist?“) oder bei potenziellen Verantwortlichen suchen.

Allgemeine Auswertungen von persönlichen Daten aus DevOps-Tools:
Bei dieser Frage sollten die Befragten angeben, ob diese sich Auswertungsmöglichkeiten von persönlichen Daten aus DevOps-Tools vorstellen könnten, welche sie selbst nicht zwangsläufig durchführen, aber von welchen sie sich vorstellen könnten, dass dies bei anderen Kollegen bzw. Unternehmen zum Einsatz kommen könnte. Hier haben alle Stakeholder angegeben, dass mit Sicherheit eine Form einer Leistungsbeurteilung bzw. -bewertung stattfinden könnte. Da in der Regel Benutzernamen und Zeitstempel in Builds, (Git) Commits und gelegentlich auch in Log-Dateien vorzufinden sind, könne man anhand dieser Daten Rückschlüsse auf die Quantität und Qualität der einzelnen Faktoren ziehen. Ein Experte hat zudem angegeben, dass in seinem eigenen Unternehmen diese Daten häufig zum Einsatz zur Problembehebung kommen: Dabei wird anhand des Benutzernamens und Zeitstempels die Person hinter einem Build, einem Git Commit oder einer anderen Ausführung ermittelt und durch gemeinsame Absprache mit diesem, eine Behebung des Problems angesteuert. Ein anderer Experte hat zusätzlich erläutert, Statistiken zum Vergleichen dieser Auswertungen mit persönlichen Daten erstellen zu können, da diese Daten unternehmensweit für nahezu jeden Mitarbeiter einsehbar sind. Durch diese genannten Ansichten der verschiedenen Stakeholder zeigt sich eine unternehmensübergreifende Präferenz zur Benutzerfreundlichkeit bzw. Einfachheit in Bezug auf die Auswertungen der persönlichen Daten in DevOps-Tools.

Persönlichen Daten in Review-Tools:
Diese Frage hat sich damit befasst, ob die Befragten Wert auf die Existenz von persönlichen Daten in Review-Tools (vgl. GitHub oder Gerrit) legen und falls ja, in was für einem Umfang sie diese benötigen bzw. nutzen.

Für fünf der insgesamt sieben Befragten war es wichtig, einen Benutzernamen in Review-Tools vorzufinden. Begründet wurde dies durch die allgemeine Möglichkeit der Zuordnung und der Identifikation eines Ansprechpartners im Falle von auftretenden Fragen und Wünschen und zum Einholen von Feedback in Bezug auf den verfassten Code. Für drei Befragte, wovon sich bereits drei den Benutzernamen gewünscht haben, möchten zusätzlich Zeitstempel in Review-Tools vorfinden können und erklären diesen Wunsch durch die Möglichkeit der zeitlichen Zuordnung der einzelnen Reviews: „Wann wurde ein Review zu meinem Programmcode bzw. zu meinem Build durchgeführt und wie relevant ist dieses im Augenblick?". Im Arbeitsalltag lässt sich also aus den Aussagen der Befragten ableiten, dass eine Einschätzung über die Dringlichkeit und Relevanz über die potenziell geforderten Änderungen und Anpassungen im Review seitens der Stakeholder erfolgt.
Die verbliebenen zwei Experten, welche keinen Wert auf persönliche Daten jeglicher Art in Review-Tools legen, begründen ihre Sichtweise dadurch, keine Reviews durchführen zu müssen bzw. keine Reviews zu erhalten oder benötigen in ihrem Arbeitsalltag lediglich Reviews und Feedback zum Inhalt ihres Codes bzw. zum Fehlschlagen von Builds.

Persönlichen Daten in Administrationstools:
Hier sollten die Befragten angeben, ob in den verwendeten Tools zur Administration persönliche Daten in Form von Zeitstempeln, Benutzernamen o.Ä. existieren. Die Befragten haben hierbei als Administrationstools angegeben, beispielsweise Jenkins als CI-Tool, SpringBoot als Framework oder einfache Kommunikationstools wie Slack und Microsoft-Teams (und insbesondere dessen Kalenderfunktion) einzusetzen. Im Prinzip sind sich alle Befragten einig, sowohl Benutzernamen, als auch Zeitstempel in Administrationstools zu benötigen und im Arbeitsalltag auch regelmäßig miteinzubeziehen. Um eine allgemeine Funktion von internen Kommunikationstools (zur eindeutigen Identifikation, mit welcher Person kommuniziert wird) zu gewährleisten und bei Builds anhand ihrer Zeitstempel (Startzeit und Dauer) und dem Autor bei Fehlschlag nachvollziehen und einen Ansprechpartner identifizieren zu können, werden hier als Gründe angegeben. Lediglich ein Experte hat angegeben, in SpringBoot auf den Einsatz von persönlichen Daten jeglicher Form verzichten zu können, da diese bisher im eingesetzten SpringBoot ohnehin nicht existent waren.

Detailgrad von Tickets in Ticketverwaltungstools:
In Ticketverwaltungstools (vgl. Jira oder Redmine2) kommen sogenannte „Issues" zum Einsatz. Damit werden die Tickets bezeichnet, welche z.T. von Kunden, aber auch von Mitarbeitern eines Softwareentwicklungsunternehmens selbst erstellt werden. Mit diesen werden beispielsweise auftretende Fehler in jeglicher Form gemeldet, Verbesserungs- bzw. Änderungswünsche angegeben oder eine andere Aufgabenstellung festgehalten. Dadurch ist es möglich, dass Mitarbeiter und Kunden einsehen, nachverfolgen und Kommentare zu den Tickets hinzufügen können, welche Probleme aufgetreten sind, welche Wünsche (z.B. von einer Applikation) gefordert sind oder welche Informationen zu einem bestimmten

1: Entwicklungstool zur agilen Softwareentwicklung, vgl. Atlassian (https://www.atlassian.com/de/software/jira)

2: Open-Source-Applikation für Projektmanagement, vgl. Redmine (https://www.redmine.org)

Thema relevant sind. Da ein grundlegendes Verständnis beim Anlegen eines solchen Issue-Tickets gewährleistet sein muss, haben alle Befragten auch angegeben, eine ausführliche Issue-Beschreibung zu benötigen. In dieser befindet sich i.d.R. der Inhalt des Tickets („Was genau muss getan werden?", der Status („Wie weit fortgeschritten ist diese Aufgabe?", ein Zeitstempel („Wann wurde diese Aufgabe erstellt?" und die Angabe eines Ticket-Erstellers und -Beauftragten („Von wem wurde diese Aufgabe angelegt und wer soll diese bearbeiten?", wobei diese bei sechs Befragten in Form eines Benutzernamen auftritt und beim verbliebenen als Vor- und Nachname.Zusätzlich dazu wünschen alle sieben Experten, dass Benutzernamen und Zeitstempel in diesen vermerkt sind. Dies begründet ein Experte dadurch, dass die Lebensdauer eines Tickets stark variieren und z.T. nach mehreren Jahren und zweistelliger Anzahl von „Assignees" (Derzeitiger Beauftragte für die Bearbeitung des Tickets) immernoch existieren kann und die Übersicht ohne diese verloren ginge - schließlich könne man so nicht nachverfolgen, welche Person zu welchem Datum bzw. zu welcher Uhrzeit was bearbeitet habe und dies für die Bearbeitung dieser Tickets erforderlich sei.

Interesse an Benutzernamen und Zeitstempeln in editierten Issue-Kommentaren und -Beschreibungen in Ticketverwaltungstools:
Ticketverwaltungstools bieten häufig die Möglichkeit, eigenen, verfassten Text, also in Kommentaren oder in der Beschreibung der Tickets, hinterher zu editieren. Zu diesem Aspekt wurden die Experten befragt, ob ein Interesse, von welcher Person zu welcher Zeit etwas (im Nachgang) editiert wurde. Hier haben sechs Befragte angegeben, Benutzernamen, Zeitstempel und die ursprüngliche Nachricht angezeigt bekommen zu wollen. Der Grund hierfür liegt erneut bei der Nachvollziehbarkeit von verfassten Kommentaren oder der Ticketbeschreibung, da es häufig vorkommt, dass Kommentare aufeinander oder auf die Beschreibung aufbauen und eine nachträgliche Änderung den Kontext dieser erschweren könnte. Vor allem bei „schwierigen Themen" sei dies häufig der Fall. Dabei reicht es drei der fünf Experten nicht aus, lediglich einen Vermerk auf einen editierten Beitrag zu erhalten - diese wünschen sich zusätzlich, den ursprünglichen Beitrag, die Uhrzeit und den Benutzernamen, vom Verantwortlichen der Änderungen, angezeigt zu bekommen, da auf diese Art und Weise relevante Informationen aus den Änderungen gewonnen werden können („Warum hat diese Person zu diesem Zeitpunkt diesen Beitrag editiert?"). Einer der sechs Experten, welche sich für editierte Beiträge interessieren, hat angegeben, keine Historie einsehen zu können und sich aufgrund dessen nicht für diese zu interessieren, weswegen eine einfache Anzeige, dass ein Beitrag editiert wurde, ausreichen würde.

Tatsächliche Aufbewahrungsdauer von Log-Dateien:
Da ein sehr großer Anteil der Befragten von über 85% (vgl. Tabelle 3.1) angegeben haben, Log-Dateien aktiv zu nutzen, ist es für den weiteren Verlauf der Analyse erforderlich, die tatsächliche Aufbewahrungsdauer der genutzten Log-Dateien im Unternehmen zu spezifizieren. Hier haben alle Befragten angegeben, dass alle Log-Dateien in der Regel für eine unbegrenzte bzw. unbestimmte Dauer aufbewahrt werden. Die Stakeholder

haben zudem meist keine Kenntnis über die genaue Aufbewahrungsdauer und beziehen sich bei ihrer Aussage auf die Existenz von sehr alten Log-Dateien (über 5 Jahre). Die Ausnahme dieser Regel betrifft manuell durchgeführte Bereinigungen von Speicher und eingeleitete Löschung spezifischer Log-Dateien: Bei einem Befragten kam es vor, dass das System einen Absturz erlitten hat, welche die Log-Dateien unbenutzbar gemacht haben, wohingegen bei einem anderen der Serverspeicher geleert werden musste und dieses Vorgehen sämtliche Log-Dateien gelöscht hat.
Aus diesen Aussagen lässt sich ableiten, dass ein Vorgehen zur automatisierten Löschung von Log-Dateien bei den Befragten bisher keine Anwendung gefunden hat.

Mögliche Verarbeitungsinteressen von persönlichen Daten in anderen Unternehmen:
Im Rahmen dieser Frage ist es relevant gewesen, die Ansichten der Experten zu möglichen Verarbeitungsinteressen von persönlichen Daten in anderen Unternehmen einzuholen. Dadurch ist es möglich, ein allgemeines Bild von Softwareentwicklungsunternehmen im Ganzen zu erhalten, falls die Ansichten der Stakeholder sich als deckungsgleich zeigen sollten. In der Tat haben etwa 57% (vier der sieben Befragten) angegeben, dass sie sich vorstellen können, wie Unternehmen die Arbeitszeiten der einzelnen Angestellten nachverfolgen können. Schließlich könne man einsehen, welche Person zu welcher Uhrzeit einen Commit bereitgestellt hat (durch Präsenz des Zeitstempels und Benutzernamens), wer im Kommunikationstool anwesend ist (Online-Status, z.B. in Microsoft Teams) oder wer zu welcher Uhrzeit auf E-Mails reagiert bzw. antwortet, behaupten die Stakeholder. Aus diesen gesammelten Daten eine Arbeitszeiten-Statistik anzulegen, sei nur einen kleinen Schritt von diesem Aspekt fortgeführt. Durch diesen Punkt könne man gebuchte Arbeitszeiten nachprüfen, korrigieren oder ggf. weglassen, da dies durch die Kontrolle redundant werden würde.
Zusätzlich hat ein Befragter angegeben, sich vorstellen zu können, dass Ressourcenmanagement betrieben werden könne: Durch die Einsicht in die Arbeits- bzw. Anwesenheitszeiten der einzelnen Angestellten, hat dieser Befragte behauptet, könne man vorhandene Ressourcen (wie z.B. eingeschaltete Server, Computer oder Strom im Allgemein) sparen, indem man diese, abhängig zu den genannten Zeiten, zur Verfügung stellt. Das jeweilige Unternehmen könne so laufende Betriebskosten sparen, was im eigenen Interesse liegen würde.

4.1.2 Kategorien mit einer Präferenz für die Privatsphäre

Persönlichen Daten in Log-Dateien:
Dieser Punkt befasst sich mit dem allgemeinen Interesse an persönlichen Daten, welche in Form von Benutzernamen, Zeitstempeln, IDs etc. in Log-Dateien sämtlicher Art auftreten können. Hier haben die Befragten einstimmig entschieden, kein Interesse an diesen zu haben und ein mögliches Entfernen in der Zukunft willkommen zu heißen. Es wurde

lediglich angegeben, dass in bestimmten Fällen (Zusammenarbeit an einem Projekt, in Testfällen von Code, Servern oder Applikationen oder auf Anfrage) eine Nachverfolgung auf Wunsch eines Kunden oder eines Auftrags erfolgen muss. Für die Befragten ist es nur wichtig, den Ablauf und auftretende Fehler von Servern, Applikationen oder Code im Allgemeinen nachverfolgen zu können - ein Interesse an bestimmten Personen oder Zeitstempeln ist nicht vorhanden, weswegen in dieser Kategorie die Privatsphäre über der Nutzerfreundlichkeit steht.

Bevorzugte Aufbewahrungsdauer von Log-Dateien:
Als Weiterleitung der Frage über die tatsächliche Aufbewahrungsdauer von Log-Dateien in Kapitel 4.1.1 wurden in dieser Kategorie die Experten dazu angehalten, ihre persönliche Präferenz zur Dauer anzugeben. Hier haben alle Befragten angegeben, eine temporäre Aufbewahrungsdauer zu bevorzugen. Begründet wurde dies dadurch, dass Log-Dateien nur für wenige bestimmte Fälle, welche i.d.R. relativ neu sind (<2 Monate), benötigt werden und über diesen Zeitraum hinaus redundant werden.

4.2 Kategorien mit abweichenden Ergebnissen

Dieses Kapitel befasst sich mit der verbliebenen Kategorie, in welcher Aussagen zu Sichtweisen seitens der einzelnen Stakeholder getroffen wurden, welche sich zu keiner eindeutigen Aussage zusammenfassen lassen und dementsprechend separat behandelt werden müssen. Als ein abweichendes Ergebnis wird dabei bezeichnet, dass bei sieben Befragten weniger als 28% die selbe Sichtweise zu einer Kategorie haben und somit kein einstimmiges Ergebnis vorliegt.

Sonstige relevante Daten aus der Sicht der Stakeholder:
Zu diesem Aspekt wurden die Experten befragt, ob ihnen sonstige, für sie relevante, Daten im Arbeitsalltag in den Sinn kommen. Diese Frage ist insofern relevant, da hierdurch potenziell neue Aspekte offenbart werden können, welche im Arbeitsalltag eines Stakeholders Anwendung findet. Dementsprechend haben lediglich drei der sieben Befragten eine individuelle Vorstellung wiedergeben können. Da diese sich selbst aber voneinander unterscheiden und keine gemeinsame Kategorie ausfindig gemacht werden konnte, werden diese im Kapitel mit den Kategorien mit abweichenden Ergebnissen behandelt: Für einen Befragten ist es beispielsweise wichtig, den Umfang eines Commits in DevOps-Tools einsehen zu können. Das bedeutet, dass eine Unterscheidung, ob es sich bei einem Commit nun um eine „kleine Fehlerbehebung“ oder eine „große Implementierung eines neuen Features“ handelt, durchgeführt werden kann. Dies sei vor allem zur Einschätzung des Arbeitsaufwandes und der Forderungen zur Herangehensweise an die Aufgabe selbst notwendig.
Ein weiterer Experte hat zusätzlich angegeben, die Überprüfung der Anwesenheit der Angestellten vor Ort sei ein wichtiger Bestandteil seines Arbeitsalltages. Als letzten Aspekt wurde die Aussage getroffen, dass die

Commit-Frequenz der eigenen Kollegen für diesen Befragten eine große Rolle spielt: Dadurch könne er, im Falle eines Merge-Requests (Anfrage zum Einbinden eines Commits in das Hauptverzeichnis, z.B. einer Applikation), einschätzen, wieviel dieser im jeweiligen Commit an Änderungen bzw. Anpassungen zu erwarten hat. Laut eigener Angabe sei es im eigenen Softwareentwicklungsunternehmen üblich, dass unterschiedliche Entwickler unterschiedliche Commit-Frequenzen aufweisen. Manche stellen deutlich größere, aber seltener Commits bereit, als manche andere, welche viele, kleine Commits als angemessen empfinden. Anhand dieser Information könne der Befragte den Commit angemessener beurteilen.

4.3 Vorgeschlagene Alternativen

Temporäre Zeitstempel in DevOps-Tools:
Zusätzlich zur gewünschten Auflösung von Zeitstempeln in DevOps-Tools haben vier der sieben Befragten angegeben, eine temporäre Lebensdauer der Zeitstempel zu bevorzugen, da diese nach einer bestimmten Zeit an Relevanz verlieren würden. Abhängig sei dies beispielsweise durch die Anzahl der zuletzt erfolgreich durchgeführten Builds, einem bestimmten Zeitraum x oder nach vollendeter Lebensdauer eines Tickets. Die Befragten haben dabei Bezug zum eigenen Arbeitsalltag genommen, in welchem die Zeitstempel unlimitiert bestehen bleiben würden und nur nach einer manuell durchgeführten Löschung des übergeordneten Speicherpunktes (z.B. eines abgelaufenen Builds, veralteten Commits etc.) verschwinden würden.

Temporäre Aufbewahrungsdauer von Log-Dateien in Abhängigkeit von Faktor xy: Zu der Frage, welche Länge zur Aufbewahrung von Log-Dateien seitens der Stakeholder bevorzugt werden würde, haben sechs von sieben Experten angegeben, eine temporäre Aufbewahrung von Log-Dateien willkommen zu heißen. Diese könne anhand der Lebensdauer von Projekten, der Lebensdauer von Programmcode, den letzten x Builds, einer festgelegten Zeit (z.B. zwei Wochen/Monaten) oder der Existenz von gemeldeten Fehlern deklariert werden. In der Regel sind sich die Befragten einig, eine Dauer von maximal wenigen Monaten zu bevorzugen. Diese Sichtweisen stellen einen Gegensatz zur tatsächlichen Aufbewahrungsdauer von Log-Dateien in Softwareentwicklungsunternehmen dar, welche in Kapitel 4.1.1 angesprochen werden. Begründet wird dieser Vorschlag durch die kaum vorhandene Nutzung von alten (z.B. >2 Monate) Log-Dateien im Arbeitsalltag der Stakeholder.

5 Diskussion der Ergebnisse

5.1 Implikationen in der Theorie

5.2 Implikationen in der Praxis

5.3 Limitation der Arbeit

6 Fazit und Ausblick

Dies ist das Kapitel, in welchem ich meine Bachelorarbeit abschließe und ein Fazit angebe.

A Further Recommendations

This appendix contains further recommendations, which our students found useful in the past.

A.1 Introduction, Related Work, and Conclusion

A.1.1 Introduction Chapter

Don't be boring! Pull in the reader with a peculiar observation or a surprising result of your thesis.

Approaches to structure your introduction:

- Start with the general, close with the specific.
- Start with what is already well-known, move on to what has only recently become known.
- State the objective of the thesis, then describe your approach.

End the introduction with a paragraph *Outline of the Thesis* that describes its structure in prose.

A.1.2 Related Work Chapter

Think about a story and which message you want to convey. Some common themes are:

- "It used to be a difficult problem, but we know how to solve it."
- "While a lot of proposals exist, none has gained traction in reality. X found that one of the primary reasons for that situation is …"
- "The field is a cat-and-mouse game between refined attacks and defenses. A common theme is … and overlooked areas are …"
- "While most of the current literature focuses on X, it would make sense to apply techniques from field Y to the problem. One could then refine the problem as …"

A.1.3 Conclusion Chapter

A basic recipe for a conclusion chapter:

- a summary,
- a critical assessment of what you have achieved,
- pointers to related other topics,
- a statement of the impact of the results, and
- pointers to future work.

A.2 Thesis Length

Typical theses range from 25 to 100 pages. Do not worry too much about the page count. Write everything that is necessary to assess and reproduce your work, but no more. Less is more!

Check the individual parts for an appropriate length: Do not write five pages in the introduction if the main part has only 15 pages.

A.3 Practices to Avoid

- Colloquial style: "Computation power has skyrocketed in the last few years."
- Widespread knowledge: "The Internet is becoming more and more important."
- Superlatives: "One of the most beautiful concepts", "great possibilities."
- Be careful with strong claims: "the only possibility", "the best solution", "impossible."

Literaturverzeichnis

[PA18] H (Pseudonym) und S Augsten. *Was ist Continuous Integration?* 2018. https://www.dev-insider.de/was-ist-continuous-integration-a-690914/ (besucht am 16. 03. 2018) (siehe S. 5).

[20a] *About*. 2020. https://git-scm.com (siehe S. 6).

[App20] Apple. *WWDC 2020 Special Event Keynote — Apple*. 2020. https://www.youtube.com/watch?v=GEZhD3J89ZE (besucht am 22. 06. 2020) (siehe S. 1).

[20b] *Auszeichnungen & Prüfsiegel*. 2020. https://www.gdata.de/ueber-g-data/auszeichnungen (siehe S. 1).

[Bar06] SB Barnes. A privacy paradox: Social networking in the United States. In: *First Monday* 11(9):(2006) (siehe S. 2).

[Bit18] Bitkom. Social-Media-Trends. In: *Bitkom Research*:(Feb. 2018), 12. https://www.bitkom.org/Presse/Anhaenge-an-PIs/2018/180227-Bitkom-PK-Charts-Social-Media-Trends.pdf (siehe S. 2).

[Bun19] Bundeskriminalamt. *Cybercrime. Bundeslagebild 2018*. Bundeskriminalamt, 2019 (siehe S. 1).

[CS20] S Chacon und B Straub. Pro Git. In:(2020), 9–11 (siehe S. 6).

[CFP04] B Collins-Sussman, BW Fitzpatrick und CM Pilato. *Version Control with Subversion*. 2004 (siehe S. 6).

[19] *Computer- und Internetnutzung im ersten Quartal des jeweiligen Jahres von Personen ab 10 Jahren*. 2019. https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Einkommen-Konsum - Lebensbedingungen/ IT - Nutzung/Tabellen/zeitvergleich-computernutzung-ikt.html (besucht am 05. 09. 2019) (siehe S. 1).

[16] *Consumer Study 2012, 2014, 2016*. 2016. https://www.nim.org/compact/fokusthemen/nicht-ohne-mein-handy-auch-jenseits-der-50/ (besucht am 06/2016) (siehe S. 1).

[Dir18] J Dirk. *Was ist Continuous Integration (Kontinuierliche Integration)?* 2018. https://www.it-talents.de/blog/it-talents/was-ist-continuous-delivery (besucht am 26. 10. 2018) (siehe S. 5).

[DB14] N Döring und J Bortz. *Forschungsmethoden und Evaluation*. Springer, 2014 (siehe S. 9, 10).

[Fle16] J Fleig. *Stakeholder erkennen und analysieren. Was sind Stakeholder und was bedeutet der Stakeholder-Ansatz?* 2016. https://www.business-wissen.de/hb/was-sind-stakeholder-und-was-bedeutet-der-stakeholder-ansatz/ (besucht am 15. 03. 2016) (siehe S. 4).

[Has15] W Hasselbring. DevOps: Softwarearchitektur an der Schnittstelle zwischen Entwicklung und Betrieb. In:(2015), 1–20 (siehe S. 4).

[HD] N Hatke und DPA. *Kundenkarten lohnen sich nicht immer*. https://www.augsburger-allgemeine.de/themenwelten/wirtschaft/Kundenkarten-lohnen-sich-nicht-immer-id7725336.html (siehe S. 2).

[IA17] Ilan_r_r und S Augsten. *Definition "Kontinuierliche Auslieferung". Was ist Continuous Delivery?* 2017. https://www.dev-insider.de/was-ist-continuous-delivery-a-664429/ (besucht am 08. 12. 2017) (siehe S. 5).

[Inc20] S Inc. *Compare Repositories*. 2020. https://www.openhub.net/repositories/compare (siehe S. 6).

[18a] *Malware-Zahlen 2017*. 2018. https://www.gdata.de/blog/2018/03/30607-malware-zahlen-2017 (besucht am 26. 03. 2018) (siehe S. 1).

[18b] *Malwarezahlen erstes Halbjahr 2018: Die Gefahr lauert im Web*. Techn. Ber. G Data Blog, 2018. /Users/samet/git/BachelorThesis/doc (besucht am 30. 08. 2018) (siehe S. 1).

[Pay19] Payback. *Umsatz der Payback GmbH in den Jahren von 2012 bis 2018*. 2019. https://de.statista.com/statistik/daten/studie/1088022/umfrage/umsatz-der-payback-gmbh/ (besucht am 09/2020) (siehe S. 2).

[Pet15] M Peters. *SVN, was sind Trunk, Branch und Co?* 2015. https://www.kussin.de/development/svn-was-

sind-trunk-branch-und-co/ (besucht am 30. 09. 2015) (siehe S. 5).

[Pet20] B Pettinger. *Recht auf Privatsphäre – ein kleiner Hoffnungsschimmer bleibt*. 2020. https://www.dr-datenschutz.de/recht-auf-privatsphaere-ein-kleiner-hoffnungsschimmer-bleibt/ (besucht am 08. 04. 2020) (siehe S. 3).

[Pic18] A Picchi. *#LogOutFacebook: The social media giant faces protests and canceled accounts*. 2018. https://www.cbsnews.com/news/logoutfacebook-the-social-media-giant-faces-naacp-protests-and-canceled-accounts/ (besucht am 18. 12. 2018) (siehe S. 3).

[Sey18] T Seyfriedt. *Stakeholder-Ansatz*. 2018. https://wirtschaftslexikon.gabler.de/definition/stakeholder-ansatz-46282/version-269567 (besucht am 15. 02. 2018) (siehe S. 4).

[Sta13] O Stallwood. Game to destroy CCTV cameras: vandalism or valid protest? In: *The Guardian*:(2013) (siehe S. 3).

[20c] *Was ist DevOps?* 2020. https://azure.microsoft.com/de-de/overview/what-is-devops/#culture (siehe S. 4).

[20d] *Was versteht man unter CI/CD?* 2020. https://www.redhat.com/de/topics/devops/what-is-ci-cd (siehe S. 6).

Eidesstattliche Erklärung

Ich erkläre hiermit gemäß § 17 Abs. 2 APO, dass ich die vorstehende Bachelorarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Bamberg, den _____ _____

Samet Murat Akcabay