



Fakultät für Informatik

Studiengang Studiengang-einsetzen

Text Text Text

Master Thesis/Bachelor Thesis

von

Max Mustermann

Datum der Abgabe: tt.mm.jjjj

Erstprüfer: Prof. Dr.

Zweitprüfer: Prof. Dr.

EIGENSTÄNDIGKEITSERKLÄRUNG / DECLARATION OF ORIGINALITY

Hiermit bestätige ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken (dazu zählen auch Internetquellen) entnommen sind, wurden unter Angabe der Quelle kenntlich gemacht.

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Rosenheim, den tt.mm.jjjj

Vor- und Zuname

Abstract

In dieser Arbeit soll mittels einer Datenerhebung von ausgewählten Open Source Projekten und einer Umfrage herausgefunden werden, welche Faktoren zum Erfolg beitragen. Was kann ein Maintainer machen um die Erfolgchancen eines Projektes zu erhöhen. Hierbei werden Faktoren analysiert wie, welchen Einfluss haben Lizenzen auf der Erfolg? ... irgendwas mit Community, Dokumentation, irgendwas mit Sponsoren....

Schlagworte:

Inhaltsverzeichnis

1	Einleitung	1
1.1	Definition von Open Source	1
1.2	Erfolg definieren	1
2	Erfolgsfaktoren	3
2.1	Lizenzen	3
2.2	Dokumentation	4
2.3	Die Community und die Projektentwicklung	5
2.4	Finanzierung und Sponsoren von Open Source	6
2.5	Beliebtheit	7
3	Datenerhebung	8
3.1	Manuelle Datenerfassung	8
3.1.1	Dokumentation	8
3.1.2	Sponsoren	8
3.1.3	Backed By	10
3.1.4	Projektarten	10
3.2	Automatisierte Datenerfassung	11
3.2.1	Daten aus der GitHub API	11
3.2.2	Daten aus der NPM API	11
3.2.3	Daten aus dem Web-Scraping	11
3.2.4	Nachbearbeitung der Daten	11
4	Ergebnisse der Datenerhebung	13
4.1	Lizenzen	13
4.2	Code of Conduct	14
4.3	Contributing Guide	15
4.4	Einfluss von Sponsoren auf den technischen Erfolg	15
5	Umfrage	16
5.1	Dokumentation	17
5.2	Beliebtheit	20
5.3	Sponsoren	21
5.4	Development	22
5.5	Freie Kategorisierung der Erfolgskriterien	23
6	Diskussion	24
6.1	Einfluss von permissiven Lizenzen auf den Erfolg	24
6.2	Der Einfluss einer guten Dokumentation auf den Markterfolg	24
6.3	Der Einfluss vom Code of Conduct und Contributing Guide auf den technischen Erfolg	25
6.4	Technischer Erfolg führt zu Markterfolg	25
6.5	Einfluss der Sponsoren auf den Erfolg	25

6.6	Zu Hypothese 9	26
7	Fazit	27
	Literaturverzeichnis	28

Abbildungsverzeichnis

2.1	VueJS Top Sponsoren	6
3.1	Web-Interface	9
3.2	Prozess der Datenerhebung	12
4.1	Effekt von Lizenzen auf GitHub Sterne	14
4.2	Effekt von Lizenzen auf Anzahl der Commits	14
4.3	Einfluss von Contributing Guide auf Mitwirkende	15
5.1	Antworten: Was zeichnet Gute Dokumentation aus?	18
5.2	Antworten: Was zeichnet Gute Dokumentation aus?	19

Tabellenverzeichnis

2.1 React vs Svelte	7
4.1 Lizenzen der erfassten Projekte	13
4.2 Relation von Code of Conduct und Beliebtheit	14
4.3 Relation von Sponsoren und Erfolg	15
5.1 Häufigste Erwähnungen der Freitexter	18
5.2 Einfluss der Beliebtheitsmerkmalen bei der Wahl von OSS	20
5.3 Einfluss von Sponsoren	21
5.4 Einfluss des Entwicklungsprozesses bei der Wahl von OSS	22

Glossar

Pull Request

Ein Pull Request ist...

(GitHub) Issues

Ein Issue ist...

Contributor

engl. Mitwirkender, Projektteilnehmer

Repository

xxx

Repositories

Code of Conduct

Contributing Guide

Contributor Covenant

1 Einleitung

Open Source ist heutzutage ein fester Bestandteil der Softwareindustrie. Von Frontend-Entwicklung über Datenbanken bis hin zu Machine Learning, überall kommen Open Source Bibliotheken, Frameworks und Programme zum Einsatz.

Im Frontend werden verschiedene Frameworks bzw. Bibliotheken wie Angular oder React verwendet, im Fall von Datenbankmanagementsysteme gibt es ebenfalls eine Vielzahl an Optionen wie PostgreSQL, MySQL oder MongoDB. Im Bereich Machine Learning werden Frameworks wie TensorFlow, Keras oder SciKit-Learn genutzt.

Doch worin unterscheiden sich die oben genannten von vielen weiteren erfolgreiche Open Source Projekte, von denen die nicht als erfolgreich definiert werden können. Mit dieser Bachelorarbeit soll die Frage beantwortet werden, *welche Faktoren Einfluss auf den Erfolg von Open Source Projekten, speziell in der JavaScript/TypeScript Umgebung haben.*

Mittels einer Datenerhebung von ausgewählten Open Source Projekten sowie einer Umfrage soll herausgefunden werden, welche Faktoren zum Erfolg eines Open Source Projektes beitragen. Ein zentraler Punkt dieser Ausarbeitung sind die extrinsischen sowie intrinsischen Anreize, die Nutzer zur Auswahl eines Produktes motivieren [Mid12]. Warum React und nicht Angular? Warum Debian und nicht Ubuntu? Aspekte wie die interne Führung und Organisation der Projekte wird hierbei nicht thematisiert.

1.1 Definition von Open Source

Diese Arbeit folgt der gleichen Definition für Open Source wie von der *Open Source Initiative*¹ definiert wird. Entsprechend werden nur Projekte betrachtet die eine von der OSI genehmigten Lizenzen² nutzt.

TODO: Lizenzen näher beschreiben, Unterscheidung zwischen Restriktiv

Unterscheidung zwischen restriktiv und permissiven Lizenzen + semi restriktiv gilt hier als permissiv! Weil....

Was macht permissive aus? - Kann uneingeschränkt benutzt werden - Kann ez verwendet werden, einfach Lizenz Text beilegen - Source Code einer MIT Lib kann in Proprietärer Software verwendet werden usw. - Beispiel: MIT, BSD, Apache Was macht GPL aus? - Kann nicht uneingeschränkt genutzt werden => one GPL Lib => Whole Programm needs to be Open Source now under gpl - GPLv2 vs GPLv3 - Was hat es dann eigentlich mit AGPL und LGPL auf sich? irgendwas mit Cloud und Server shit MPL - Die Mitte zwischen GPL und MIT So under the terms of the MPL, it allows the integration of MPL-licensed code into proprietary codebases, but only on condition those components remain accessible

1.2 Erfolg definieren

Der Erfolg wird in der Literatur häufig in verschiedene Bereiche unterteilt. Im Artikel von Midha und Palvia wird zwischen *Markterfolg* und *technischem Erfolg* unterschieden. Midha et al. definieren Markterfolg als Grad des Nutzerinteresses an einem Projekt, welches sich in der Beliebtheit des Projektes widerspiegelt.

¹ <https://opensource.org/osd>

² <https://opensource.org/licenses/category>

1 Einleitung

Den technischen Erfolg definieren Midha et al. durch die Entwickleraktivität, d.h. durch den Aufwand, den die Entwickler für das Projekt betreiben beispielsweise die Häufigkeit und Frequenz von Updates und neuen Versionen [Mid12].

Im Artikel von Steward et al. wird zwischen *Nutzerinteresse* und *Entwickleraktivität* unterschieden [Ste06]. Subramaniam et al. gehen sogar weiter und unterteilen den Erfolg in *Nutzerinteresse*, *Entwicklerinteresse* und *Projektaktivität* [Sub09].

Diese Bereiche werden getrennt betrachtet, da verschiedene Faktoren unterschiedlichen Einfluss auf den Erfolg eines Projektes haben. Laut Midha et al. und Steward et al. wirkt sich wachsendes Interesse der Nutzern positiv auf den Markterfolg aus, während sich die Entwickleraktivität positiv den technischen Erfolg auswirken [Mid12, Ste06].

TODO: Warum genau diese Metriken? Sterne, Downloads etc.???

- Kein Copy Paste von Autoren, weil die nicht GH verwendet haben (Nicht aktuell) - Beliebte vergleichsmetriken (I guess lol)

In dieser Arbeit werden die Erfolgsfaktoren betrachtet, die zum Markt- bzw. technischen Erfolg eines Projektes beitragen. Der Markterfolg wird anhand folgender Metriken gemessen: Downloads und Sterne auf GitHub. Der technische Erfolg wird anhand von Metriken wie: Anzahl der Commits und Anzahl der Mitwirkenden am Projekt gemessen.

2 Erfolgsfaktoren

Im folgenden Kapitel werden verschiedenen mögliche Erfolgsfaktoren mittels Literatur analysiert und basierend darauf Hypothesen abgeleitet. In späteren Kapiteln werden diese Hypothesen anschließend mittels einer Datenerhebung und Umfrage geprüft.

2.1 Lizenzen

Laut Subramaniam et al. spielen Lizenzen eine signifikante Rolle für den Erfolg von Open Source Software. Freizügige Lizenzen wie MIT oder BSD haben einen positiven Einfluss vor allem auf Softwareentwickler. Denn Entwickler nutzen OSS, um es in eigene Projekte einzubauen, gegebenenfalls zu modifizieren und ein Endprodukt mit der OSS Komponente weiterzuverbreiten. Das ist mit restriktiven Lizenzen wie GPL meist nicht bedingungslos umsetzbar. Restriktive Lizenzen wie GPL wirken sich daher negativ bis neutral auf den Erfolg von OSS aus. Wenn die Software allerdings an Endnutzer gerichtet ist, wie zum Beispiel die Instant-Messaging-Dienst Telegram¹, spielt die Lizenz eine weniger wichtige Rolle, da Weiterverbreitung und Modifizierung für diese Nutzergruppe keine Rolle spielen [Sub09].

Stewart et al. widerspricht der zweiten Aussage von Subramaniam et al. laut ihnen haben permissive Lizenzen nicht nur auf das Entwicklerinteresse, sondern auch auf das Nutzerinteresse einen positiven Einfluss. Während restriktive Lizenzen einen nicht signifikanten Einfluss auf Entwickleraktivität hätten. [Ste06]

Laut Midha und Palvia wirken sich permissive Lizenzen positiv auf den Markterfolg aus, allerdings nur zu Beginn eines Projektes. Restriktive Lizenzen wiederum wirken sich negativ auf den technischen Erfolg aus [Mid12].

Meine Hypothese ist, dass sich offene Lizenzen durchgängig positiv auf ein Projekt auswirken. Offene Lizenzen werden tendenziell eher von Unternehmen verwendet als Projekte mit restriktiven Lizenzen, das führt zu einem dazu, dass die Beliebtheit und Bekanntheit des Projektes steigt, als auch die Wahrscheinlichkeit dass die Unternehmen zum Open Source Projekt etwas beitragen oder Sponsoren werden.

H 1. *Offene Lizenzen haben positiven Einfluss auf den Markterfolg.*

Steigt die Beliebtheit eines Projekts, so steigt auch das Interesse von Open Source Entwickler an einem renommierten Projekt mitzuwirken.

H 2. *Offene Lizenzen haben positiven Einfluss auf den technischen Erfolg.*

¹ <https://telegram.org/>

2.2 Dokumentation

TODO: Was ist gute Dokumentation? Quellen Suchen. Gute Dokumentation definieren Nach dem Zweiten Satz idealer w

Dokumentationen spielen eine entscheidende Rolle für den Erfolg eines Projekts. Ohne eine gute Dokumentation ist die Software für die Benutzer als auch Mitwirkender schwer zugänglich. Mailing Listen und StackOverflow können als eine Ergänzung zur Dokumentation dienen, allerdings kann diese dadurch nicht ersetzt werden [Ban13].

Laut XY ist gute Dokumentation...

Eine gute Dokumentation ist auch ein Mittel, um neue Nutzer zu gewinnen. Im Artikel von Dagenais et al. heißt es, dass schon das Vorhandensein eines *Getting Started* Tutorials, den Nutzer beim Entscheidungsprozess positiv beeinflussen kann [Dag10]. Ist man als Nutzer die ersten Schritte mit einer neuen Programmiersprache, Framework oder Bibliothek gegangen, steigt die Wahrscheinlichkeit, dieses Produkt auch zu nutzen. Beispielsweise hat die Web-Bibliothek ReactJS² auf der Homepage simple Beispiele, die man live editieren kann, ohne sich vorher etwas downloaden zu müssen, um einen ersten Eindruck von React zu gewinnen. Des Weiteren gibt es ein sehr ausführliches Tutorial³ welches über 5 Kapitel mit 21 Unterkapitel alle Grundbausteine von React abdeckt, um das gesamte Feature-Set in Kürze zu präsentieren und neue Entwickler von React zu überzeugen.

Laut einer GitHub Umfrage im Jahr 2017 sind unvollständige oder verwirrende Dokumentationen das größte Problem für Open Source Nutzer. Eine gute Dokumentation hingegen lädt nicht nur neue Nutzer ein, sondern kann auch Nutzer zu Mitwirkenden machen. Sei es durch das Erstellen von Issues oder eines ersten Pull Requests. Hierfür spielen vor allem Contributing Guides und ein Code of Conduct eine wichtige Rolle, hierzu mehr im nächsten Kapitel [Git17].

Dokumentationen spielen eine wichtige Rolle, um neue Nutzer als auch neue Mitwirkende für ein Projekt zu gewinnen. Daher wird die Hypothese aufgestellt, dass Projekte mit guter Dokumentation, einen höheren Markterfolg haben.

H 3. *Gute Dokumentationen ziehen mehr Nutzer an und führen so zu einem höheren Markterfolg.*

² <https://reactjs.org/>

³ <https://reactjs.org/docs/getting-started.html>

2.3 Die Community und die Projektentwicklung

Ein weiterer Erfolgsfaktor der OSS ist die Community, die sowohl aus Nutzern als auch von OSS-Entwickler besteht. Es liegt in der Verantwortung der Projektleiter die Community aufzubauen und zu pflegen [Ban13, Gita].

In einem Leitfaden von GitHub wird die Bedeutung der Community und wie diese ihr Aufbau. Hierbei ist vor allem die Rede von einem Code of Conduct und Contributing Guide [Gita]. Das Ziel ist es eine offene und wachsende Community aufzubauen, die dem technischen Erfolg dienen soll.

Das *Code of Conduct* ist ein Dokument, welches die Erwartungen an das Verhalten der Projektteilnehmer festlegt. Das Übernehmen und Durchsetzen des Code of Conducts kann dazu beitragen, eine positive und soziale Atmosphäre für alle zu schaffen [Gita]. Häufig findet sich hier im Root-Verzeichnis des Projektes die Datei `CODE_OF_CONDUCT.md`. Als Vorlagen dient in der Regel das *Contributor Covenant*⁴.

Der *Contributing Guide* ist eine Einführung, für neue Mitwirkende, die sich an dem jeweiligen beteiligen wollen. Hier finden sich die Anleitungen und Vorlagen für Bug Reports, Feature Requests, vorgehen bei Pull Requests, sowie Richtlinien bezüglich Coding Styles und Testabdeckung [Gita]. Einige Projekte beginnen ihre `CONTRIBUTING.md` mit einem Dank an den Leser und künftigen Mitwirkenden, wie beispielsweise Chakra-UI⁵, mit den Worten *"Thanks for showing interest to contribute to Chakra UI, you rock!"*. Somit wird die Hypothese aufgestellt, dass Projekte mit einem Code of Conduct bzw. Contributing Guide einen höheren technischen Erfolg haben.

H 4. *Das Vorhandensein eines Code of Conduct führt zu einem höheren technischen Erfolg.*

H 5. *Das Vorhandensein eines Contributing Guides führt zu einem höherem technischen Erfolg.*

Da der Entwicklungsprozess in einem Open Source Projekt offen einsehbar ist, kann dieser von potenziellen Nutzern auch bewertet werden. Metriken wie Aktualität des letzten Commits, Regelmäßigkeit der Commits, Anzahl der aktiven Maintainer oder die Antwortzeit der Entwickler auf Tickets könnten hierbei für einen Nutzer interessant sein. Die sechste Hypothese ist daher, dass Projekte mit hohem technischen Erfolg einen positiven Einfluss auf die Wahl der OSS haben.

H 6. *Technischer Erfolg führt zu Markterfolg.*

⁴ <https://www.contributor-covenant.org/>

⁵ <https://github.com/chakra-ui/chakra-ui>

2.4 Finanzierung und Sponsoren von Open Source

Ein weiterer Faktor, der Nutzer davon überzeugen kann ein gewisses Framework oder eine Bibliothek gegenüber einer anderen zu nutzen, ist die finanzielle Sicherheit eines Projekts. Es wird vermutet, dass Projekte mit Sponsoren beliebter bei Nutzern sind als Projekte ohne. Die Information, ob ein Projekt Sponsoren hat, ist für einen neuen potenziellen Nutzer hierbei in der Regel sehr leicht ersichtlich. VueJS beispielsweise macht sowohl auf der Website als auch in der README.md auf die Sponsoren aufmerksam und bedankt sich für ihre Unterstützung, wie in Abbildung 2.1 zu erkennen ist.

Daraus folgt die siebte Hypothese,

H 7. *Sponsoren haben einen positiven Einfluss auf die OSS-Auswahl, demnach haben gesponserte Projekte einen höheren Markterfolg.*

Gleichzeitig kann die finanzielle Unterstützung der Sponsoren die Produktivität fördern. Die Projekte werden schneller weiterentwickelt und sorgfältiger gewartet. Daraus folgt die achte Hypothese:

H 8. *Gesponserte Projekte haben einen höheren technischen Erfolg.*

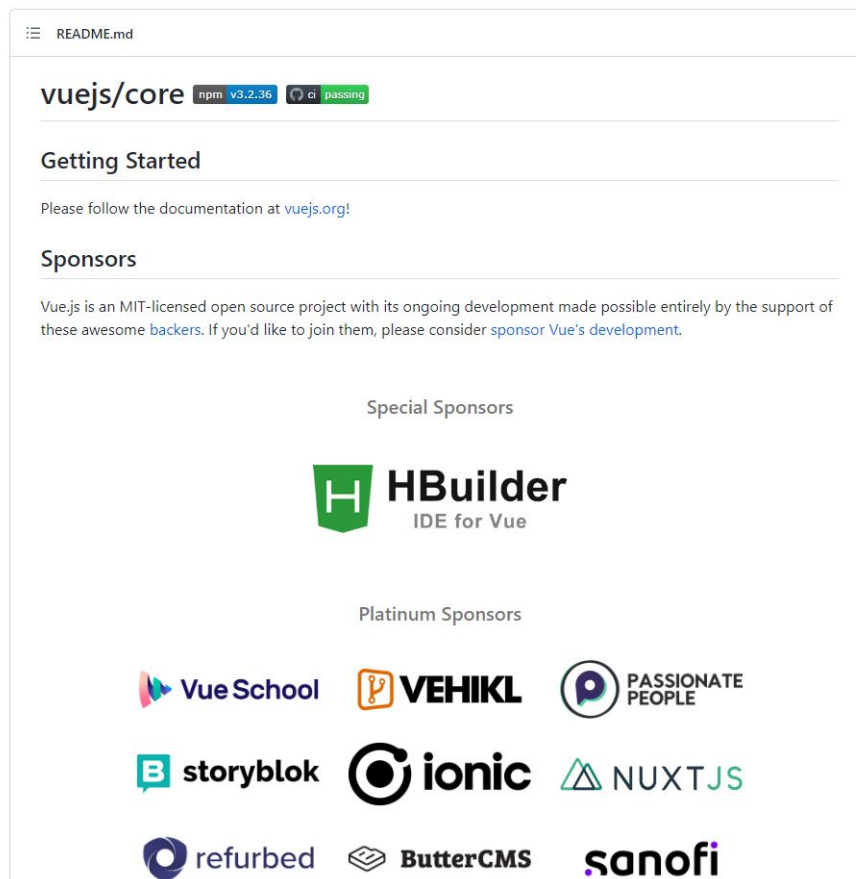


Abbildung 2.1 VueJS Top Sponsoren

2.5 Beliebtheit

Des Weiteren kann die Beliebtheit eines Projekts für die Nutzer eine wichtige Rolle spielen. Midha et al. fanden eine starke Korrelation zwischen der vergangenen Beliebtheit eines Projekts, und der aktuellen Beliebtheit. Der Grund hierfür sei, dass die Beliebtheit als Entscheidungskriterium bei der Auswahl eines Projektes verwendet wird [Mid12]. In dieser Arbeit wird die Beliebtheit anhand der Anzahl der GitHub Sterne und Downloads gemessen.

Subramaniam et al. sprechen von vom sogenannten *Netzwerkeffekt*, dieser wirkt sich laut [Sub09] positiv auf den Erfolg von OSS aus. Der Begriff Netzwerkeffekt kommt aus der Volkswirtschaftslehre und beschreibt ein Phänomen, bei dem ein Produkt oder eine Dienstleistung einen zusätzlichen Wert erhält, wenn mehr Menschen diesen nutzen. Im Softwareumfeld wird der Netzwerkeffekt in, z.B. Anzahl an Tutorials oder Beiträge auf StackOverflow zeigen. Das wiederum schafft einen stärkeren Anreiz das beliebtere Tool zu nutzen. Im Vergleich von ReactJS und Svelte wie in der Tabelle 2.1 dargestellt wird, zeigt sich, dass ReactJS etwas mehr als 3-mal so viele Sterne auf GitHub hat, aber über 100-mal so viele Beiträge auf StackOverflow. Daraus ergibt sich die Hypothese:

H 9. *Beliebte Projekte werden von Nutzern bei der Wahl von OSS bevorzugt.*

Tabelle 2.1 React vs Svelte

	GitHub Stars	npm downloads	Tutorials ^a	StackOverflow Fragen
React	186k	14.6 mio	438 mio	380k
Svelte	57k	278k	2.3 mio	3k

^a Anzahl der Google Ergebnisse

3 Datenerhebung

TODO: Bessere Title ? Datenerhebung Github

Das Ziel dieser Datenerhebung ist es die Hypothese H1, H2, H4, H5, und H8 zu prüfen.

Hier wurden 108 Open Source Projekte auf GitHub ausgewählt. Die Studie von Midha et al. hat sich auf die Programmiersprachen C++ beschränkt, ähnlich wird sich auch diese Arbeit auf JavaScript (JS) und TypeScript (TS) beschränken. Grund hierfür ist, dass sich verschiedene Programmiersprachen schlecht miteinander vergleichen lassen [Mid12]. JavaScript wurde für die Datenerfassung ausgewählt, da es zurzeit die Sprache mit den meisten Projekten auf GitHub ist. TS wird mit aufgenommen, da es ein Obermenge von JS ist. Es ist unmöglich reine JS Projekte [aufzufinden](#).

Erhoben wurden nur Software Projekte. Repositories wie E-Books, Tutorials oder Lehrplattformen wurden ausgeschlossen. Projekt Archive wurden ebenfalls ausgeschlossen

3.1 Manuelle Datenerfassung

Mithilfe der GitHub Suchfunktion¹ wurden JS und TS Projekte ausgewählt. Die Ausgabe wurde nach Anzahl Sternen sortiert wiedergegeben und in dieser Reihenfolge erfasst. Die Ausgabe der ersten Suche beinhaltete überwiegende Mehrzahl der Projekte mit MIT Lizenz. Um eine höhere Diversität des Lizenzen-Milieus zu gewährleisten wurde im Laufe der Datenerfassung explizit, nach nicht MIT lizenzierten Projekten gefiltert. Zum Erfassen der Daten wurde ein selbst entwickeltes Web-Interface verwendet, um die Erhebung zu vereinfachen. Wie in Abbildung 3.1 zu sehen ist wurden zunächst Identifikationsdaten notiert. Für GitHub wurde `<owner>/<project>` verwendet, wie sie auch in der URL oder auf der Project Page zu finden sind. Für NPM wurde der Package Name verwendet, falls dieser vorhanden war. Applikationen beispielsweise haben in den meisten Fällen kein NPM Package. Des Weiteren wurden auch Niveau der Dokumentation, Vorhandensein von Sponsoren, wer hinter dem Projekt steht und um was für ein Typ Projekt es sich handelt gesammelt. Alle Einträge wurden an ein NodeJS Backend gesendet und als CSV abgespeichert. Im den nächsten Unterkapiteln wird näher erläutert, wie die Kriterien, die von Hand erfasst wurden zustande gekommen sind und welche Bedeutung die Werte in der CSV-Datei haben.

3.1.1 Dokumentation

Die Dokumentation des Projektes wurde analysiert und nach folgenden Kriterien kategorisiert:

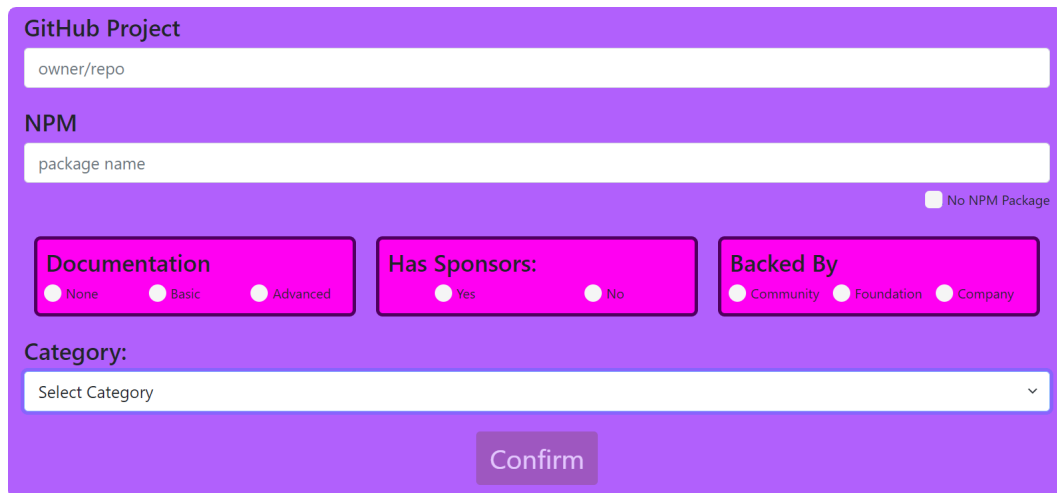
- 0 = keine Dokumentation
- 1 = basis Dokumentation, rein textuell
- 2 = Dokumentationen mit Demos oder Live Beispielen, Einführungsvideos oder ähnliches

3.1.2 Sponsoren

Für jedes Projekt wurde geprüft, ob es Sponsoren hat. Die Anzahl an Sponsoren bzw. die Einnahmen durch Sponsoren wurden nicht beachtet. Das Vorhandensein der Sponsoren wurde folgend codiert:

¹ <https://github.com/search/advanced>

3 Datenerhebung



The image shows a web interface for data collection, titled "GitHub Project". It features a form with several input fields and radio button groups. The "GitHub Project" section has a text input for "owner/repo". The "NPM" section has a text input for "package name" and a checkbox for "No NPM Package". Below these are three radio button groups: "Documentation" with options "None", "Basic", and "Advanced"; "Has Sponsors:" with options "Yes" and "No"; and "Backed By" with options "Community", "Foundation", and "Company". A "Category:" section has a dropdown menu labeled "Select Category". At the bottom is a "Confirm" button.

GitHub Project

owner/repo

NPM

package name

☐ No NPM Package

Documentation

☐ None ☐ Basic ☐ Advanced

Has Sponsors:

☐ Yes ☐ No

Backed By

☐ Community ☐ Foundation ☐ Company

Category:

Select Category

Confirm

Abbildung 3.1 Web-Interface

0 = hat keine Sponsoren

1 = hat Sponsoren

3.1.3 Backed By

Im nächsten Schritt wurde vermerkt wer hinter einem Projekt steht, hierbei wurde in drei Kategorien eingeteilt:

- 0 = Eines reinen Community Projektes unabhängig von Unternehmen. Beispiel: VueJS.
- 1 = Von einer Stiftung wie OpenJS² unterstütztes oder entwickeltes Projekt.
Beispiel: NodeJS.
- 2 = Projekte die von Unternehmen entwickelt werden, wie React von Facebook beispielsweise.

Diese Information fanden sich entweder auf der GitHub-Page, Homepage des Projektes oder das Unternehmen ist der Besitzer des Repositories.

3.1.4 Projektarten

Im letzten Schritt wurde das Projekt in eins der folgenden Kategorien zugeordnet:

- | | |
|---------------|------------------|
| • Utility | • Framework |
| • UI | • Test-Framework |
| • Application | • Open-Core |
| • Library | • API |

Erläuterung der Kategorien

- **Utility**, wurden Projekte kategorisiert, die nicht direkt in Projekte eingebaut werden, sondern als Tool verwendet werden. Beispiel: shelljs/shelljs³
- **Application**, sind eigenständige Produkte wie draw.io⁴

² <https://openjsf.org/>

³ <https://github.com/shelljs/shelljs>

⁴ <https://github.com/jgraph/drawio>

3.2 Automatisierte Datenerfassung

Im zweiten Teil der Erhebung wurden weitere Daten automatisiert gesammelt. Hierfür wurde die GitHub API⁵, NPM API⁶ und für einige Daten Web-Scraping verwendet.

Zu diesem Zweck wurde eine weitere NodeJS Applikation geschrieben, welche die CSV-Datei aus dem Kapitel 3.1 ausliest und mithilfe der APIs und Web-Scraping eine neue CSV-Datei generiert. In Abbildung 3.2 wird der Prozess der Datenerhebung grafisch dargestellt. In den folgenden Unterkapiteln wird aufgelistet welche Daten mittels welcher Methode gesammelt wurden.

3.2.1 Daten aus der GitHub API

Mittels der GitHub API wurden folgende Daten gesammelt:

- Anzahl der GitHub-Sternen
- Erstellungsdatum des Repositories
- Vorhandensein einer *CODE_OF_CONDUCT.md*
- Vorhandensein einer *CONTRIBUTING.md*
- Lizenz
- Anzahl der Commits in den letzten 12 Monaten
- Anzahl der Issues in den letzten 12 Monaten

3.2.2 Daten aus der NPM API

Mit der NPM API wurden die Downloads der letzten 7 Tage abgefragt.

3.2.3 Daten aus dem Web-Scraping

Mithilfe von Web-Scraping wurden folgende Daten erfasst:

- Die Anzahl von *UsedBy* auf der GitHub Page des Projektes.
- Anzahl der Gesamt-Commits auf dem default Branch
- Anzahl der Contributor

Anmerkung:

Ein Mitwirkender zählt nur dann als solcher, wenn dessen Commit entweder auf dem *default* oder *gh-pages* Branch liegt. Die Commits auf anderen Branches werden nur dann gezählt, wenn ein merge auf einen der vorherig erwähnten Branches stattfindet. Gleiches gilt für Commits [GHa].

3.2.4 Nachbearbeitung der Daten

Nachdem erheben der Daten war eine Nachbearbeitung notwendig. Einige Projekte hatten die Lizenz *other*. Der Grund hierfür ist, dass die *LICENSE.md* nicht dem Standardtext einer Lizenz entsprochen hat. Ein Beispiel wäre die Lizenz von *meteor*⁷, welche nach dem offiziellen Lizenztext noch eine Anmerkung bezüglich benutzter Bibliotheken hat. Alle als *other* gekennzeichneten Projekte wurden manuell geprüft und korrigiert. Zudem wurden Projekte ohne Lizenzen oder Lizenzen die nicht von der *Open Source*

⁵ <https://docs.github.com/en/rest>

⁶ <https://github.com/npm/registry>

⁷ <https://github.com/meteor/meteor/blob/devel/LICENSE>

3 Datenerhebung

Initiative zugelassene sind komplett aussortiert. Nach dem gleichen Prinzip wurden Projekte kontrolliert, die laut API kein Contributing Guide bzw. kein Code of Conduct haben. Grund hierfür ist, dass GitHub das Vorhandensein dieser Dateien nur dann erkennt, wenn diese im Hauptverzeichnis liegen und exakt CONTRIBUTING.md bzw. CODE_OF_CONDUCT.md heißen. Einige Projekte hatten Tippfehler in den Dateinamen, oder den Contributing Guide bzw. Code of Conduct war Teil der README.md statt eine eigene Datei bzw. auf der Website des Projektes.

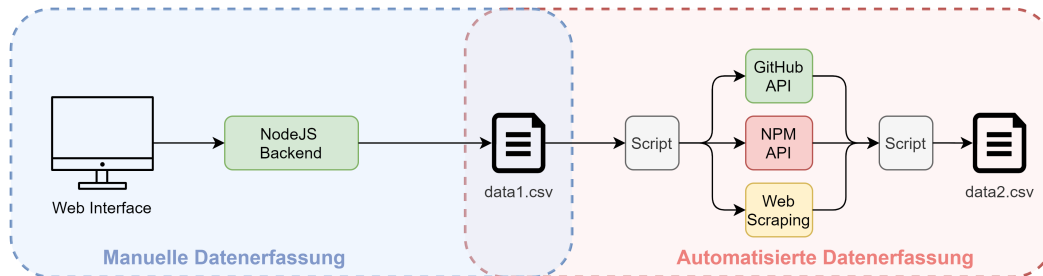


Abbildung 3.2 Prozess der Datenerhebung

4 Ergebnisse der Datenerhebung

TODO: Kleine Einleitung + 108 Projekte erwähnen.

In den folgenden unterkapitel... werden die Ergebnisse der 108 Projekte ...

4.1 Lizenzen

TODO: Verweis auf Kapitel 2.1 Lizenzen => Warum genau diese Aufteilung

Um die Hypothese H1 zu prüfen wurden der Datensatz in Projekte mit freizügigen und restriktiven Lizenzen. Die Aufteilung erfolgt hierbei, wie im Kapitel 1.3.3.7 zu Lizenzen näher erklärt wurde. In der Tabelle 4.1 findet sich eine Zusammenfassung aller Lizenzen der Projekte die erfasst wurden. 91% der Projekte haben eine freizügige Lizenz, wobei die MIT Lizenz mit 69% die beliebteste aller Lizenzen ist.

Um die Gruppen miteinander zu vergleichen wurde der Median der Anzahl der Sterne verwendet, da diese sehr stark verteilt ist. Der Median für permissive Lizenzen liegt bei 27.515 Sternen, für restriktive bei 21.595,5. Somit haben Projekte mit permissiven Lizenzen im Median 27,4% mehr Sterne als Projekte mit restriktiven Lizenzen.

Aufgrund der großen Mengenunterscheiden zwischen freizügigen und restriktiven Projekten wurde zusätzlich ein Ranking erstellt. Wie in Abbildung 4.1 zu sehen ist wurden hierfür die Projekte nach Sternen sortiert. Von 108 ausgewerteten Projekten liegt das erste mit restriktiver Lizenz auf Platz 35, das heißt die Top 31% der Projekte sind alle permissiv lizenziert. Projekte mit freizügigen Lizenzen übertreffen die restriktiven sowohl in der Menge als auch in der Beliebtheit.

Um die zweite Hypothese zu prüfen wurden ebenfalls die Mediane verglichen, allerdings wurden hier Anzahl der Mitwirkenden sowie Commits der letzten 12 Monate betrachtet. Der Median wurde in diesem Fall gewählt, da sowohl die Anzahl Commits als die der Mitwirkenden starke Ausreißer haben.

Im Median haben restriktive Projekte 78 Mitwirkende und 74,5 Commits, permissive Projekte haben 274,5 Mitwirkende und 182 Commits. In einen weiterem Ranking, sortiert nach Mitwirkenden befindet sich das erste Projekt mit restriktiver Lizenz auf Platz 51. Im Ranking der Anzahl der Commits erreicht, wie man in Abbildung 4.2 sehen kann, das erste restriktive Projekt Platz 9.

Im Ranking der Anzahl der Commits befinden sich unter den ersten 20 Projekten zwei OSP mit restriktive Lizenzen.

Tabelle 4.1 Lizenzen der erfassten Projekte

Permissive Lizenz	Anz.	Restriktive Lizenz	Anz.
MIT	74	AGPLv3	4
Apache 2.0	18	GPLv3	3
BSD 3-Clause	3	LGPLv2.1	1
BSD 2-Clause	2	OSLv3.0	1
ISC	1	LGPLv3	1
Gesamt	98	Gesamt	10

4 Ergebnisse der Datenerhebung

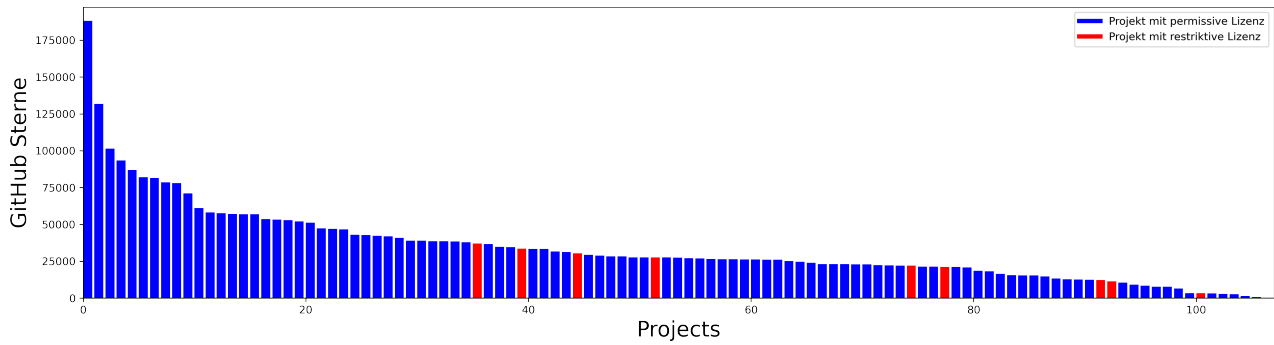


Abbildung 4.1 Effekt von Lizenzen auf GitHub Sterne

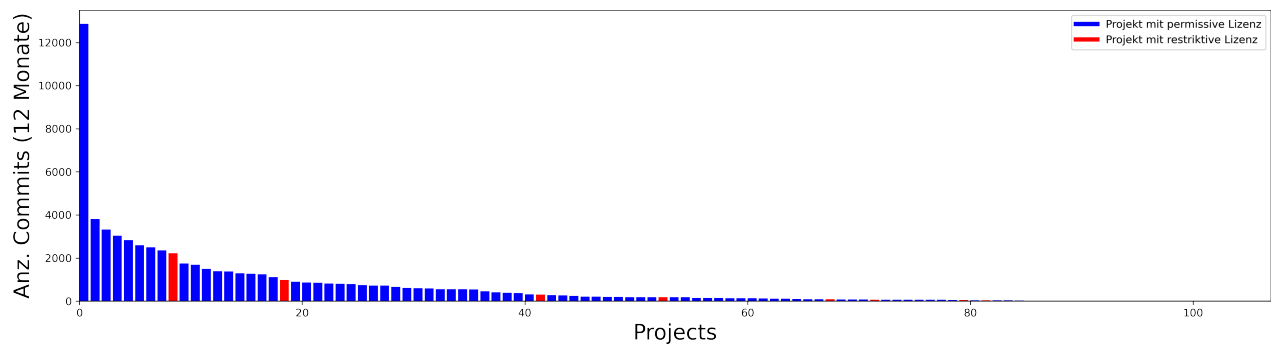


Abbildung 4.2 Effekt von Lizenzen auf Anzahl der Commits

4.2 Code of Conduct

Um Hypothese H4 zu testen wird der Datensatz erneut in zwei Gruppen geteilt, Projekte mit Code of Conduct und Projekte ohne. Verglichen wurden Anzahl der Commits und Mitwirkenden mittels Median.

Wie in Tabelle 4.2 zu erkennen ist, haben 53% der Projekte einen Code of Conduct. Im Vergleich beider Gruppen haben Projekte mit Code of Conduct 717% mehr Commits und 201% mehr Mitwirkender, als Projekte ohne.

Tabelle 4.2 Relation von Code of Conduct und Beliebtheit

	Projekte mit Code of Conduct	Projekte ohne Code of Conduct
Anzahl	57	51
Median Commits	556	68
Median Mitwirkende	313	104

4.3 Contributing Guide

Wie im Kapitel 4.2 wurde auch hier der Datensatz in zwei Gruppen unterteilt. Projekte mit und ohne einen Contributing Guide (CG). Um Hypothese H5 zu testen soll der Zusammenhang zwischen dem Vorhandensein eines CG und Anzahl der Mitwirkenden bzw. Commits erforscht werden.

Anders als beim Code of Conduct, haben hier 86% aller Projekte einen CG. Verglichen wurden erneut der Median der Anzahl der Mitwirkenden und Anzahl der Commits, Projekte mit einem CG haben 463% mehr Commits und 412% mehr Mitwirkende.

Zusätzlich werden die Projekte nach Anzahl der Mitwirkenden (siehe Abbildung 4.3) sortiert, wie man erkennen kann haben die Projekte mit den meisten Mitwirkenden alle einen CG. Das erste Projekt ohne CG belegt hier Platz 54. Nach dem gleichen Schema wurden auch nach Anzahl der Commits sortiert, das erste Projekt ohne CG belegt Platz 42. In beiden verglichen platzieren sich Projekte mit CG weit vor Projekten ohne.

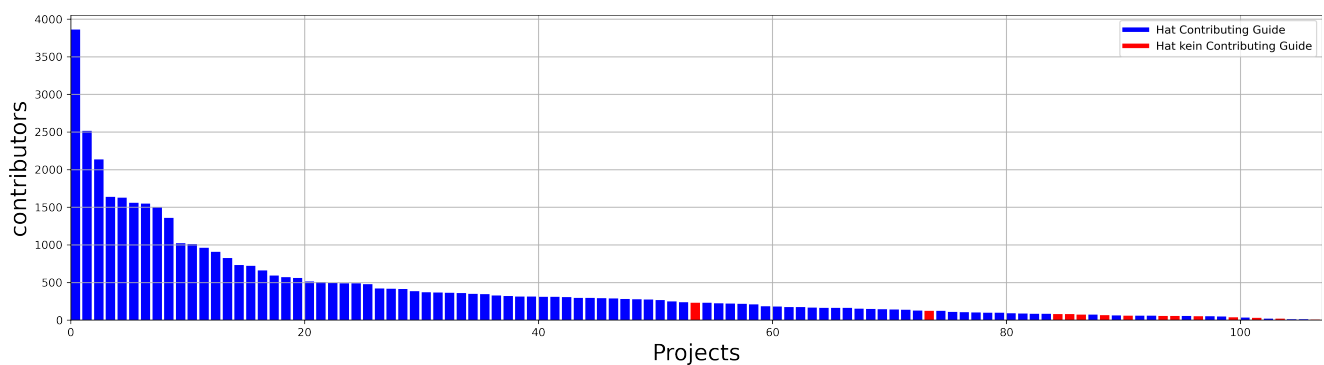


Abbildung 4.3 Einfluss von Contributing Guide auf Mitwirkende

4.4 Einfluss von Sponsoren auf den technischen Erfolg

Erneut wurde der Datensatz in zwei Gruppen geteilt, Projekte mit und ohne Sponsoren. Verglichen wurde der Median der Commits der letzten 12 Monate sowie die Anzahl an Mitwirkenden (Siehe Tabelle 4.3). 52% der Projekte haben Sponsoren. Projekte mit Sponsoren haben 72% mehr Commits und 86% mehr Mitwirkende

Tabelle 4.3 Relation von Sponsoren und Erfolg

	Projekte mit Sponsoren	Projekte ohne Sponsoren
Anzahl der Projekte	56	52
Median Commits der letzten 12 Monate	221	128
Median Anz. Mitwirkenden	289	155

5 Umfrage

Als Ergänzung zur Datenerfassung der GitHub Projekte wurde auch eine Umfrage durchgeführt. Ziel der Umfrage war es herauszufinden worauf die Nutzer von Open Source achten, wenn sie ein Projekt wählen. Das übergeordnete Ziel war es die Hypothesen H3, (H6), H7 und H9 zu prüfen. An der Online-Umfrage haben Softwareentwickler der adesso SE, sowie Studenten der TH Rosenheim teilgenommen. Insgesamt erhielt die Umfrage 308 Antworten. Zur Erstellung der Umfrage wurde *cryptpad.org* verwendet [Cry].

Hierbei sollten die Teilnehmer Aussagen zu den Thema Dokumentation, Beliebtheit, Sponsoren und Entwicklung bewerten. Als Bewertungsskala wurde eine 5-Punkte Likert-Skala verwendet [Lik32].

Interpretation der Durchschnitte: 1 = der Teilnehmende stimmt der Aussage gar nicht zu, 5 = der Teilnehmende stimmt der Aussage vollkommend zu. 3 = Undecided, doesnt matter etc.

5.1 Dokumentation

Im ersten Teil der Umfrage mussten die Teilnehmenden zu zwei Aussagen verschiedene Kriterien bewerten. Diese Kriterien wurden gewählt basierend auf Erfahrung als Entwickler als auch Beobachtungen vieler Dokumentationen während der manuellen Datenerfassung zu aus Kapitel 3.1.1.

Die erste Aussage war *"Wie wichtig sind Ihnen folgende Aspekte der OSS-Dokumentation"*, mit den Punkten:

- Übersichtlichkeit
- Einfache Sprache
- Live-Demos
- Übersetzungen (z.B. Englisch -> Deutsch)
- Das Vorhandensein einer Getting Started Page
- Code Beispiele
- Strukturierung der Dokumentation

Die zweite Aussage war *"Die folgenden Punkte würden mich von der Nutzung eines OSS-Projekts abhalten oder mich möglicherweise dazu veranlassen, nach Alternativen zu suchen."*, mit den Punkten:

- Keine Code Beispiele
- Schlecht strukturierte Dokumentation
- Dokumentation zu komplex
- Fehlende Übersetzung (z.B. fehlende deutsche Übersetzung)
- Fehlende Getting Started Page

Die Aussagen wurden auf einer 5-Punkte Skala bewertet, hierbei entspricht 1 *Stimme gar nicht zu* und 5 *Stimme vollkommen zu*. Um die Aussagen miteinander zu vergleichen, wurde jeweils der Mittelwert berechnet.

Die zwei wichtigsten Aspekte einer Guten Dokumentation sind laut den Befragten, Code Beispiele und Übersichtlichkeit in beiden Fällen mit einem durchschnittlichen Wert von 4,36. Gefolgt von guter Struktur (4,09) und das Vorhandensein eines *Getting Started* (3,98). Einfache Sprache (3,08) und Live-Demos spielt für die Befragten eine weniger wichtige Rolle. Übersetzungen belegen mit 1,53 den letzten Platz und spielt somit die unwichtigste Rolle in einer guten Dokumentation.

Die zweite Aussage war *"Die folgenden Punkte würden mich von der Nutzung eines OSS-Projekts abhalten oder mich möglicherweise dazu veranlassen, nach Alternativen zu suchen."* Am höchsten bewertet wurde der Punkt *Fehlende Code Beispiele* (4,07) und ist somit der wichtigste Teil einer guten Dokumentation, gefolgt von schlechter Struktur (3,91). Eine komplexe Dokumentation oder das Fehlen einer *Getting Started* Seite wurde hierbei als mittelmäßig kritisch eingestuft, mit einer durchschnittlichen Bewertung von 3,29 bzw. 3,10. Das Fehlen einer Übersetzung ist mit 1,37 das unwichtigste Kriterium. In den Abbildungen 5.1 und 5.2 wird das Meinungsbild der Teilnehmer grafisch dargestellt.

Zusammenfassend kann man sagen, dass Code Beispiele ein Must Have einer Dokumentation ist. Das Vorhandensein wird als wichtig eingestuft, während das Fehlen dazu führen könnte, dass Nutzer zu alternativen greifen würden. Übersetzungen hingegen werden als gleichgültig betrachtet, das Vorhandensein bietet wenig Mehrwert, die Abwesenheit wird nicht als kritisch betrachtet.

Beide Fragen hatten jeweils ein Freitextfeld, indem die Teilnehmenden die Möglichkeit hatten weitere Aspekte einer guten bzw. schlechten Dokumentation zu nennen. Diese Freitextfelder waren optional und wurden von 95 bzw. 71 der gesamten 308 Teilnehmenden genutzt. Alle Einträge wurden kategorisiert und zusammengefasst.

In der Tabelle 5.1 finden sich die am häufigsten genannten Punkte. Aktualität, Vollständigkeit und UX waren die am meisten genannten Eigenschaften bezüglich guter Dokumentation, die jeweiligen Pendants,

5 Umfrage

veraltete Dokumentation, unvollständige Dokumentation und schlechte UX waren die häufigsten genannten Gründe, um ein OSS Projekt nicht zu nutzen. Diese drei Merkmale vervollständigen somit die vorhin erwähnten *Must-Haves* einer guten Dokumentation.

Tabelle 5.1 Häufigste Erwähnungen der Freitexter

Gute Dokumentation	Erwähnungen	Schlechte Dokumentation	Erwähnungen
Aktualität	28	Veraltet	25
Vollständigkeit	21	Unvollständig	15
Gute UX	14	Schlechte UX	11
Versionierung	10	Fehlerhaft	7
Changelog vorhanden	6	Keine Doku vorhanden	6
Gute Code Beispiele	5	Code Beispiele funktionieren nicht	5

Anmerkung:

Im Fall von UX wurden folgende Punkte zusammengefasst: Suchfunktion/Verlinkung, Design und Übersichtlichkeit.

Die abschließende Frage zum Thema Dokumentation war: "Hat eine schlechte Dokumentation Sie jemals dazu gebracht, ein alternatives Projekt zu wählen?". 81% der Teilnehmenden haben diese Frage mit "Ja" beantwortet.

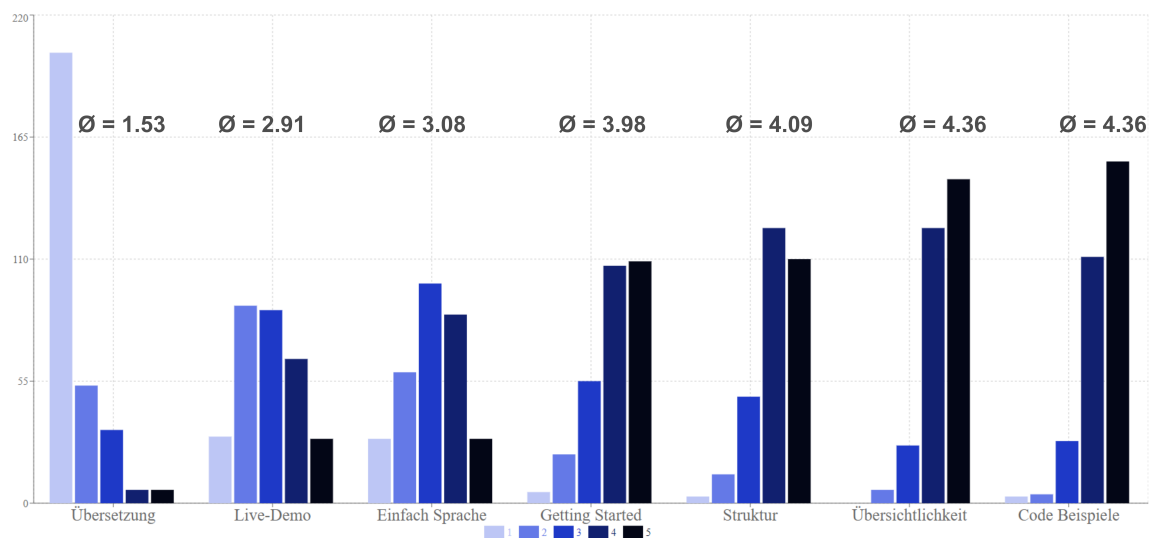


Abbildung 5.1 Antworten: Was zeichnet Gute Dokumentation aus?

5 Umfrage

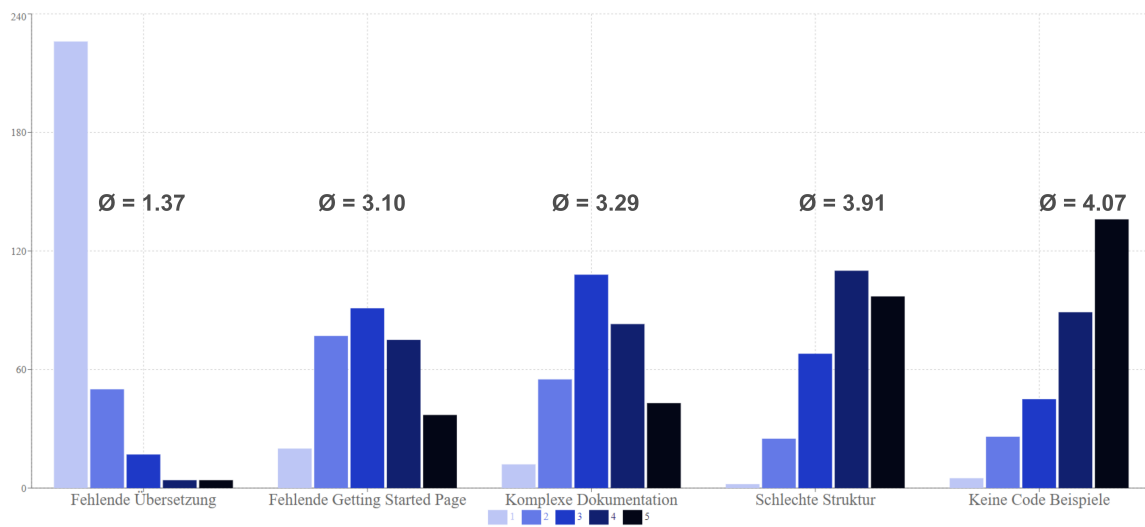


Abbildung 5.2 Antworten: Was zeichnet Gute Dokumentation aus?

5.2 Beliebtheit

Im zweiten Teil der Umfrage, soll geklärt werden, welchen Einfluss die Beliebtheit, bei der Wahl eines Projektes hat. Wie zuvor auch sollten die Teilnehmer Kriterien auf einer 5-Punkte Skala bewerten. Die Mittelwerte der Antworten finden sich in Tabelle 5.2 wieder.

Die Aussage war *"Wie sehr achten Sie bei der Auswahl von OSS auf..."*, mit den Punkten:

- Anzahl der GitHub Sterne
- Anzahl der Mitwirkenden eines Projektes
- Anzahl von Sponsoren
- Trends
- Anzahl an Fragen und Antworten auf StackOverflow

Diese Punkte wurden gewählt, da es klassischen Vergleichsmerkmalen sind, um die Beliebtheit von OSS zu vergleichen. GitHub Sterne und Downloads sind eine schnelle und einfache Methode, um die aktuelle Beliebtheit eines Projektes zu vergleichen. Trends hingegen stellen GitHub Sterne oder Downloads über Zeit dar, häufig genutzte Tools für die Analyse von Trends sind *StackOverflow Trends*¹, *npm trends*² oder *Google Trends*³.

Anders als bei der Dokumentation gibt es hier kein Kriterium mit starker Zustimmung. Die Downloads sind hierbei mit einer durchschnittlich Zustimmung von 3,27 die beliebteste Vergleichsmetrik für Beliebtheit gefolgt von Sternen (2,92). Anzahl der Mitwirkenden (2,68), StackOverflow Fragen und Antworten (2,60) sowie Trends (2,23) werden tendenziell weniger beachtet. Die Anzahl der Sponsoren wird mit einer durchschnittlichen Bewertung (1,7) fast gar nicht beachtet.

Die letzte Frage zum Thema Beliebtheit war *"Hat die geringe Popularität eines Projekts Sie jemals davon abgehalten, es zu nutzen?"* 54% der Teilnehmer haben die Frage mit *"Nein"* beantwortet. Die leichte Ablehnung der Frage deckt sich mit den vorherigen Ergebnissen. Nutzer achten auf Beliebtheit weniger als auf die Qualität der Dokumentation bei der Wahl von OSS.

Tabelle 5.2 Einfluss der Beliebtheitsmerkmalen bei der Wahl von OSS

Downloads	Sterne	Mitwirkende	StackOverflow Fragen/Antworten	Trends	Sponsoren
3.27	2.92	2.68	2.60	2.23	1.7

¹ <https://insights.stackoverflow.com/trends>

² <https://www.npmtrends.com/>

³ <https://trends.google.de/trends/>

5.3 Sponsoren

TODO: Fliegt evtl ganz raus?

Im dritten Teil der Umfrage ging es darum herauszufinden welchen Einfluss Sponsoren bei der OSS Wahl spielen. Auch hier wurde mit der 5-Punkte Skala bewertet, mit der Frage: *"Bewerten Sie folgende Aussagen."* Die Aussagen waren:

1. Projekte mit Sponsoren wirken zukunftssicherer
2. Ich bevorzuge es, wenn möglich, Projekte mit Sponsoren zu nutzen
3. Gesponserte Projekte sind meistens qualitativ besser

Tabelle 5.3 Einfluss von Sponsoren

Aussage:	1.	2.	3.
	3.27	2.92	2.68

Umformulieren, Ausformulieren... Wie in Kapitel 5.2 dargelegt, wird auf das Vorhandensein von Sponsoren wenig geachtet. Hier soll es aber darum gehen wie das Vorhandensein von Sponsoren empfunden wird. Am ehesten wird die Präsenz von Sponsoren mit zukunftssicherer verbunden. Qualitativ werden werden gesponserte Projekte nicht als besser empfunden. *Meine Vermutung liegt hier in der Tatsache, das nicht gesponserte Projekte ebenfalls Qualitativ hochwertig sein können/sind. Somit hängt Qualität nicht von Geld ab...*

Wie in Kapitel 5.2 dargelegt, wird auf das Vorhandensein von Sponsoren weniger geachtet. Hier erkennt man womöglich den Grund dafür. Gesponserte Projekte werden weder bevorzugt, noch als qualitativ hochwertiger betrachtet, sprich es wird auf Sponsoren nicht geachtet.

5.4 Development

Im vorletzten Teil der Umfrage ging es um den Einfluss des Entwicklungsprozesses bei der Auswahl von OSS. Die Frage war: *„Wie sehr achten Sie auf die folgenden Punkte, wenn Sie ein Projekt wählen?“*

- Anzahl aktiver Maintainer
- Ticket/Issue Verhältnis Open/Closed
- Antwortzeit der Entwickler auf Tickets/Issues
- Regelmäßigkeit der Commits
- Aktualität des letztes Commit

Diese Punkte wurden gewählt, da es leicht einsehbare Daten sind, die den Entwicklungsprozess eines Projektes widerspiegeln.

Ähnlich wie bei der Dokumentation legen die Nutzer einen große Wert auf Aktualität (vgl. Tabellen 5.1) Mit einer durchschnittlichen Zustimmung von 3,94 ist diese das wichtigste Kriterium des Entwicklungsprozesses, gefolgt von Anzahl der Maintainer (3,33), Regelmäßigkeit der Commits (3,28), Verhältnis von offenen und abgeschlossenen Tickets (2,90) und Antwortzeit der Entwickler auf Tickets (2,86).

Tabelle 5.4 Einfluss des Entwicklungsprozesses bei der Wahl von OSS

Aktualität	Anzahl der Maintainer	Regelmäßigkeit	Verhältnis	Antwortzeit
3.94	3.33	3.28	2.90	2.86

5.5 Freie Kategorisierung der Erfolgskriterien

Im letzten Teil der Umfrage sollten die Teilnehmer verschiedene Kriterien per *Drag and Drop* sortieren. Die Fragestellung lautete *Sortieren Sie nach den für Sie wichtigsten Kriterien bei der Auswahl von OSS.* mit folgenden Punkten zum Sortieren

- Gute Dokumentation
- Beliebtheit
- Anzahl von offenen Issues/Tickets
- Trends
- Projekt hat Sponsoren

Aufgrund der Übersichtlichkeit hat sich die Auswahl hier auf fünf beschränkt. Es wurde versucht alle vorherigen Themen abzudecken. Je nach Platzierung, haben die Kriterien Punkte bekommen, 5 Punkte für den ersten Platz, 4 für den zweiten usw. der letzte Platz bekam einen Punkt.

Das wichtigste Kriterium ist Gute Dokumentation mit 1329 Punkten, gefolgt von Beliebtheit (1165), Anzahl der offenen Issues/Tickets (794), Trends (624), Projekt hat Sponsoren (559).

6 Diskussion

Ziel dieser Arbeit war es....

6.1 Einfluss von permissiven Lizenzen auf den Erfolg

In Kapitel 2.1 wurde die Hypothese aufgestellt, dass permissive Lizenzen einen positiven Effekt auf den Markterfolg haben. Um diese Hypothesen zu Prüfen wurde eine Datenerhebung mit 108 Projekten durchgeführt. Als Metrik des Markterfolges wurden GitHub Sterne verwendet. Wie die Ergebnisse aus Kapitel 4.1 zeigen, haben Projekte mit permissiven Lizenzen 27,4% mehr Sterne zusätzlich sind die Top 32% Projekte permissiv lizenziert. Und das trotz expliziter Suche nach restriktiven Projekten, wie in Kapitel 3.1 beschrieben. Bei der Manuellen Datenerfassung war es zusätzlich notwendig explizit nach restriktiven Projekten zu suchen, um den Anteil von restriktiver Projekte zu erhöhen. Das zeigt zum einen, dass permissive Lizenzen allgemein bevorzugt werden und permissiv lizenzierte Projekte auch allgemein erfolgreicher sind. Somit bestätigt sich Hypothese H1. Ursprünglich sollten auch die NPM Downloads als Beliebtheitsmerkmalen verwendet und verglichen werden. Allerdings sind nur 4 der 10 restriktiven Projekte auf NPM.

Des Weiteren wurde die Hypothese aufgestellt, dass permissive Lizenzen einen positiven Einfluss auf den Technischen Erfolg haben. Zum Prüfen der Hypothese wurden wie in Hypothese 1 vorgegangen, allerdings wurden hier Anzahl der Commits in den letzten 12 Monaten und Anzahl der Mitwirkenden verglichen. Projekte mit permissiven Lizenzen haben im Median sowohl mehr Commits (252%) als auch mehr Mitwirkende (144%). Somit bestätigt sich die Hypothese H2, permissive Projekte sind technisch erfolgreicher.

TODO: Sieh .tex

Dieser Arbeit hat sich nur mit Projekten beschäftigt, die hauptsächlich in JavaScript/Typescript programmiert sind. Das JS/TS Umfeld ist hierbei dominiert von permissiven Lizenzen, dies hat zur Folge gehabt, dass von den 108 Projekten nur 10 Projekte restriktiv lizenziert sind. Diese Ungleichverteilung erschwert Aussagen bezüglich Lizenzen.

Die Ergebnisse bestätigen die Studie von Subramaniam et al. in dem Punkt, dass permissive Lizenzen einen positiven Effekt auf den Erfolg von Projekten haben [Sub09]. Über den Effekt von restriktiven Lizenzen lassen sich hier keine starken Aussagen treffen, aufgrund kleiner Datenmengen mit restriktiven Lizenzen.

6.2 Der Einfluss einer guten Dokumentation auf den Markterfolg

Im Kapitel 2.2 wurde die Hypothese aufgestellt, dass gute Dokumentationen mehr Nutzer anziehen würden und somit zu einem höheren Markterfolg führen würde. Um diese Hypothese zu prüfen wurde eine Umfrage durchgeführt. Die Umfrage hatte 308 Teilnehmer. Die Teilnehmenden sollten auf einer Likert Skala Aussagen zustimmen oder ablehnen, hier hatte das Thema Dokumentation die höchste Zustimmung verglichen mit den anderen Themengebieten der Umfrage, dies zeigt ein allgemein hohes Interesse am Thema Dokumentation. Diese Aussage wird vom Fakt untermauert, dass wie in Kapitel 5.5 gezeigt wird,

die Dokumentation das wichtigste Kriterium, bei der Auswahl von OSS ist. Des Weiteren gaben 81% der befragten an, aufgrund schlechter Dokumentation schon mal zu einem alternativen Projekt gewechselt zu haben. Das zeigt die Wichtigkeit einer guten Dokumentation und bestätigt somit die Hypothese H3.

Die Freitext Felder haben gezeigt das die Kriterien die es zum bewerten gab nicht vollständig waren. Vor allem Aktualität und Vollständigkeit sind weitere wichtige Eigenschaften für Dokumentationen. Für weitere Forschung wäre daher sinnvoll die Kriterien die es zu bewerten gilt zu erweitern. Dies würde auch dazu führen das man die Wichtigkeit dieser Kriterien vergleichbarer macht.

In einer Umfrage der Open Source Survey aus 2017 stellte sich heraus, dass unvollständige bzw. verirrende Dokumentation als größtes Problem in Open Source gelten [Git17]. Diese Arbeit bestätigt diese Ergebnisse, und erweitert diese. Neben Komplexität und Unvollständigkeit sind die wichtigste Punkte einer Guten Dokumentation, Code Beispiele und Aktualität.

6.3 Der Einfluss vom Code of Conduct und Contributing Guide auf den technischen Erfolg

Im Kapitel 2.3 wurden zwei Hypothesen aufgestellt, basierend auf den Open Source Guide von GitHub aufgestellt [Gita]. Die Hypothese H4 beschäftigt sich mit dem Effekt des *Code of Conducts* auf den technischen Erfolg. Hypothese H5 hingegen beschäftigt sich mit dem Effekt des *Contributing Guides* auf den technischen Erfolg. Zum prüfen beider Hypothesen wurden die Daten aus der Datenerhebung verwendet. Das Code of Conduct haben 53% der 108 analysierten Projekte. Einen Contributing Guide 86%.

Gemessen wurde der technische Erfolg anhand Commits und Anzahl der Mitwirkenden. Projekte mit einem Code of Conduct haben deutlich mehr Commits (717%) und Mitwirkende (201%). Projekte mit einem Contributing Guide haben 463% mehr Commits und 412% mehr Mitwirkende als Projekte ohne. Damit sind beide Hypothese eindeutig belegt.

Eine plausible Erklärung für den starken unterschied, kann damit begründet werden, dass es sich für kleine Projekte nicht lohnt eine Code of Conduct einzuführen, da die Größe der Teams zu klein ist.

Im Fall des Contributing Guides, könnte die ungleiche Verteilung in den Daten die Ursache sein.

Es zeigt sich allerdings deutlich, dass die Empfehlungen von GitHub auch durchgesetzt werden, vor allem in großen Projekten.

6.4 Technischer Erfolg führt zu Markterfolg

Ich glaub diese Hypothese schmeiß ich wieder aus da ich die nicht ordentlich belegen/wiederlegen kann

6.5 Einfluss der Sponsoren auf den Erfolg

Im Kapitel 2.4 wurde die Hypothese aufgestellt, dass Sponsoren einen positiven Einfluss bei der Wahl von OSS hätten. Die Hypothese wurde mittels Umfrage getestet. Wie in Kapitel 5.3 und 4.4 bereits gezeigt, spielen Sponsoren für die Nutzer von OSS fast gar keine Rolle.

Somit wird Hypothese 6 nicht unterstützt.

Bezüglich Sponsoren wurde zusätzlich noch die Hypothese aufgestellt, dass Sponsoren sich positiv auf den technischen Erfolg auswirken würden. Diese Hypothese wurde mittels der Datenerhebung geprüft. Wie sich herausstellt haben Projekte mit Sponsoren 72% mehr Commits und 86% mehr Mitwirkende.

6.6 Zu Hypothese 9

Die Hypothese H9, Beliebte Projekte werden von Nutzern bei der Wahl von OSS bevorzugt. Wird nur schwach/Nicht unterstützt? (54% Nein etc....)

In Kapitel 5.2 sieht man das fast die Hälfte schon auf Beliebtheit achtet, aber halt erstens nur die Hälfte und zweites kommt es auf die Vergleichsmetrik an. Mehr als die Hälfte zumindest achtet auf Downloads.... (Schwach unterstützte Hypothese? Garnicht? idk)

Gleiches war bei den Contributor nicht umsetzbar, da die GitHub API keinen Endpoint hierfür hat, somit werden Contributor über den gesamten Zeitraum des Projektes verglichen. (Commits nur der letzten 12 Monate)

- Umfrage hatte nur den Umfang von 308 Teilnehmern und kann deshalb nicht generalisiert werden...
- Die Daten aus dem *Crawler* sind nur ein Snapshot der aktuellen Lage, sprich man kann nur sagen. Ja es gibt eine Korrelation zwischen z.B. Sponsoren und User Interest, aber nicht was was hervorruft. Hat das Projekt Sponsoren weil es viele Nutzer hat *oder* hat das Projekt viele Nutzer weil es Sponsoren hat? Dafür bräuchte man eine Datenerhebung over time. Diese ist allerdings im kleinen Zeitfenster einer Bachelorarbeit nicht machbar. [Mid12] hat beispielsweise in einem Zeitfenster von 3 Jahren die Datenerhebung gemacht.

7 Fazit

Note

Das Fazit ist: Zusammenfassend, kurz, bündig, Gesamtdarstellung, keine Beispiele oder neuen Informationen

- Das Fazit beantwortet die Forschungsfrage
- Die wichtigsten Ergebnisse werden dargestellt
- Keine neuen Informationen oder Interpretationen
- Keine Beispiele oder Zitate
- Die Arbeit reflektiert und kritisch hinterfragt
- Schließt mit einem Ausblick ab

Forschungsfrage war, was macht Open Source erfolgreich, (evtl etwas spezifizieren) In dieser Arbeit wurde herausgearbeitet welche Aspekte von Open Source Einfluss auf dessen Erfolg haben und welche nicht.

Literaturverzeichnis

- [Ban13] W. Bangerth und T. Heister. What Makes Computational Open Source Software Libraries Successful? *Computational Science & Discovery*, 6(1):015010, Nov. 2013.
- [Cry] Cryptpad.org. <https://cryptpad.org/>. Zuletzt aufgerufen am 05.06.2022.
- [Dag10] B. Dagenais und M. P. Robillard. Creating and Evolving Developer Documentation: Understanding the Decisions of Open Source Contributors. In *Proceedings of the Eighteenth ACM SIGSOFT International Symposium on Foundations of Software Engineering*, FSE '10, S. 127–136. Association for Computing Machinery, New York, NY, USA, Nov. 2010.
- [GHa] Why Are My Contributions Not Showing up on My Profile? <https://docs.github.com/en/account-and-profile/setting-up-and-managing-your-github-profile/managing-contribution-graphs-on-your-profile/why-are-my-contributions-not-showing-up-on-my-profile#contributions-not-showing-up-on-my-profile-if-my-commit-was-not-made-in-the-default-or-gh-pages-branch>. Zuletzt aufgerufen am 18.05.2022.
- [Gita] Building Welcoming Communities. <https://opensource.guide/building-community/>. Zuletzt aufgerufen am 02.04.2022.
- [Gitb] Starting an Open Source Project. <https://opensource.guide/starting-a-project/>. Zuletzt aufgerufen am 07.04.2022.
- [Gitc] Your Code of Conduct. <https://opensource.guide/code-of-conduct/>. Zuletzt aufgerufen am 07.04.2022.
- [Git17] Open Source Survey. <https://opensourcesurvey.org/2017/>, 2017. Zuletzt aufgerufen am 03.04.2022.
- [Lik32] R. Likert. A technique for the measurement of attitudes. *Archives of psychology*, 1932.
- [Mid12] V. Midha und P. Palvia. Factors Affecting the Success of Open Source Software. *Journal of Systems and Software*, 85(4):895–905, Apr. 2012.
- [Ste06] K. J. Stewart, A. P. Ammeter und L. M. Maruping. Impacts of License Choice and Organizational Sponsorship on User Interest and Development Activity in Open Source Software Projects. *Information Systems Research*, 17(2):126–144, 2006.
- [Sub09] C. Subramaniam, R. Sen und M. L. Nelson. Determinants of Open Source Software Project Success: A Longitudinal Study. *Decision Support Systems*, 46(2):576–585, Jan. 2009.