1 Einleitung

TODO: Überleitung zur Fragestellung überarbeiten.

In der heutigen Software-Entwicklung ist Open Source ein fester Bestandteil der Software Industrie. Von Frontend-Entwicklung über Datenbanken bis hin zu Machine Learning, überall kommen Open Source Bibliotheken, Frameworks und Software zum Einsatz. Im Frontend werden Frameworks bzw. Bibliotheken wie Angular oder React verwendet, bei der Auswahl von Datenbankmanagementsysteme stehen einem eine Vielzahl von Optionen zur Auswahl wie PostgreSQL, MySQL oder MongoDB und im Bereich Machine Learning werde Frameworks wie TensorFlow, Keras oder SciKit-Learn genutzt.

Doch worin unterscheiden sich diese und viele andere Erfolgreiche Open Source Projekte, von nicht so weit verbreiteten Wettbewerber. 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 etc.? 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 Lizenzen[2] nutzt.

1.2 Erfolg definieren

In der Literatur wird häufig in verschiedene Bereichen des Erfolgs unterteilt. Im Artikel von Midha und Palvia wird zwischen *Markterfolg* und *technischem Erfolg* unterschieden. Markterfolg definiert Midha et al. als Grad des Nutzerinteresses an ein Projekt, welches sich in der Beliebtheit des Projektes widerspiegelt. Den technischen Erfolg definiert 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].

In Steward et al. wird zwischen *Nutzerinteresse* und *Entwickleraktivität* unterschieden [Ste06]. Subramaniam et al. geht sogar weiter und unterteilt in *Nutzerinteresse, Entwicklerinteresse und Projektaktivität* [Sub09].

1 https://opensource.org/osd
2 https://opensource.org/licenses/category

Diese Bereiche werden getrennt betrachtet, da verschiedene Faktoren unterschiedlichen Einfluss auf den Erfolg eines Projektes haben. Während sich wachsendes Interesse bei Nutzern positiv auf den Markterfolg auswirkt, wirkt sich die Entwickleraktivität positiv den technischen Erfolg aus [Mid12, Ste06].

TODO: Footnote ausformuliren ggf link auf Glossar **TODO:** Was sind GitHub Sterne?

In dieser Arbeit werden die Erfolgsfaktoren betrachtet, die zum Markt- bzw. technischen Erfolg eines Projektes beitragen. Der Markterfolg wird anhand Metriken gemessen wie: Downloads, Sterne auf GitHub sowie die Anzahl der Nutzer. Der technische Erfolg wird anhand von Metriken wie: Anzahl der Commits, Anzahl der Mitwirkenden am Projekt, Geschwindigkeit in der Tickets abgearbeitet werden und Verhältnis zwischen offenen/abgeschlossenen Issues[3] gemessen.

3 Was sind Issues

2 Erfolgsfaktoren

Es gibt viele Faktoren, die Einfluss auf den Erfolg eines Projektes haben. In diesem Kapitel werden die Faktoren in zwei Klassen aufgeteilt und näher erklärt. Die *Haupterfolgsfaktoren* werden in dieser Bachelorarbeit mittels Datenerhebung und Umfrage analysiert. *Weitere Faktoren* werden zusätzlich auf Basis von Literatur betrachtet und diskutiert. Zu den Haupterfolgsfaktoren gehören Eigenschaften wie *Lizenzen, Qualität, Dokumentation, Community, Sponsoren* und *Network Effekt*. Diese Faktoren alleine schaffen aber kein Gesamtbild des Erfolgs, deshalb wird im Kapitel **??** auf zusätzlich Aspekte eingegangen. Diese werden allerdings nicht mittels Datenerhebung oder Umfrage analysiert, da der Aufwand diese empirisch zu erfassen zu hoch wäre, stattdessen findet die Analyse auf rein literarischer Ebene statt. Hierzu gehören Faktoren wie *Das richtige Timing, Modularität und Komplexität, Responsibility Assignment*.

2.1 Haupterfolgsfaktoren

TODO: Einleitender Satz für Kapitel 2.1

2.1.1 Lizenzen

Laut Subramaniam et al. spielen Lizenzen eine signifikante Rolle für den Erfolg von Open Source Software. Freie Lizenzen wie MIT oder BSD haben einen positiven Einfluss vor allem auf Software Entwickler. 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 Chat-App Telegram[1], 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 nicht-restriktive-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 freie Lizenzen positiv auf den Markterfolg aus, allerdings nur zu Beginn eines Projektes. Restriktive Lizenzen wiederum wirken sich negativ auf den technischen Erfolg aus [Mid12].

Ich glaube, 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 zum einen dazu, dass die Beliebtheit und Bekanntheit des Projektes steigt, als auch die Wahrscheinlichkeit das die Unternehmen zum Open Source Projekt etwas beitragen oder Sponsoren werden.

1 https://telegram.org/

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.*

2.1.2 Gute Dokumentation

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

Eine gute Dokumentation ist auch ein Mittel, um neue Nutzer zu gewinnen. In 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 nutzten. Beispielsweise hat die Web-Bibliothek ReactJS[2] 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 Tutorial[3] 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 Contributor 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 Dokumentationen, erfolgreicher sind. Sowohl im Bereich des Markt- als auch technischen Erfolgs.

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

2.1.3 Eine Community Aufbauen

Ein weiterer Erfolgsfaktor in OSS ist die Community, sowohl die Community von Nutzern, als auch die Community von OSS-Entwickler. Es liegt in der Verantwortung der Projektleiter die Community aufzubauen und zu pflegen. [Ban13, Gita]. Auf GitHub finden sich Standardmäßig zwei Dateien, welche auf die Wahrnehmung dieser Verantwortung hindeutet. Die `CODE_OF_CONDUCT.md` und `CONTRIBUTING.md`.

Das `CODE_OF_CONDUCT.md` ist ein Dokument, welches die Erwartungen an das Verhalten der Projektteilnehmer festlegt. Das Übernehmen und Durchsetzten des Code of Conducts kann dazu beitragen, eine positive und soziale Atmosphäre für alle zu schaffen [Gitc]. Häufig werden hierfür Vorlagen von der *Contributor Covenant* Website[4] verwendet.

2 https://reactjs.org/
3 https://reactjs.org/docs/getting-started.html
4 https://www.contributor-covenant.org/

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

Die `CONTRIBUTING.md` Datei ist eine kurze Einführung, für potenzielle neue Contributor, wie man am jeweiligen Projekt mitwirken kann. Hier finden sich Anleitungen und Vorlagen für Bug Reports, Feature Requests, vorgehen bei Pull Requests, sowie Richtlinien bezüglich Coding Styles und Testabdeckung [Gitb]. Einige Projekte beginnen ihre `CONTRIBUTING.md` mit einem dank an den Leser und künftigen Contributor, wie beispielsweise Chakra-UI[5], mit den Worten *"Thanks for showing interest to contribute to Chakra UI, you rock!"*. Somit wird die Hypothese aufgestellt, dass Projekte mit einem Contributing Guide eine höhere Chance für technischen Erfolg haben.

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

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

2.1.4 Sponsoren

Ein weiterer Faktor, der Nutzer davon überzeugen kann ein gewisses Framework oder Bibliothek gegenüber einer anderen zu nutze, kann das Vorhandensein von Sponsoren sein. Den Sponsoren wird meist ein Abschnitt in der README.md oder auf der Website des Projektes gewidmet, hier bedanken sich die Maintainer bei den jeweiligen Sponsoren. Häufig werden Logos der Firmen oder Profilbilder der Einzelpersonen aufgelistet. Somit ist es schnell ersichtlich ob ein Projekt gesponsert ist oder nicht.

Wenn Nutzer sehen, dass ein gewisses Framework oder eine Bibliothek von Unternehmen oder Einzelpersonen gesponsert werden, kann die Wahrnehmung der Qualität des Produktes verstärkt wird, was sich im Endeffekt dann im Markterfolg widerspiegelt.

H 6. *Sponsoren haben einen positiven Einfluss bei der Auswahl von Nutzern, gesponserte Projekte haben eine höhere Wahrscheinlichkeit auf Markterfolg.*

Sponsoren garantieren auch, dass ein Projekt auch noch morgen existiert, weiter aktiv gewartet wird und neue Features entwickelt werden können. Somit wirken sich Sponsoren auch positiv auf den technischen Erfolg aus.

H 7. *Gesponserte Projekte haben einen höheren technischen Erfolg*

Anmerkung

Man muss allerdings anmerken, dass einige Projekte von Unternehmen entwickelt werden. Diese finanzieren ihre Projekte meist selbst und sind somit nicht abhängig von Sponsoren. React beispielsweise kommt aus dem Hause Facebook, während VueJS Unabhängig ist und auf Sponsoren angewiesen ist. In diesem Kontext zählen Projekte von Unternehmen auch als gesponsert.

2.1.5 Netzwerkeffekt

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 nutzten. Im Software Umfeld würde sich der Netzwerkeffekt in, z.B. Anzahl an Tutorials oder Fragen/Antworten auf StackOverflow zeigen.

Ist ein Projekt erfolgreich werden Entwickler mehr Fragen stellen und Tutorials verfassen, was wiederum einen stärkeren Anreiz schafft das beliebtere Tool zu nutzten. Vergleicht man beispielsweise ReactJS und Svelte wird man feststellen das React etwas mehr als 3-mal so viele Sterne auf GitHub hat, aber über 100-mal so viele Fragen auf StackOverflow.

	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

TODO: Schöner Formulieren

Midha et al. fand 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].

Subramaniam et al. spricht von vom sogenannten *Network Effekt*, dieser wirkt sich laut [Sub09] positiv auf den Erfolg von OSS aus.

Projekte die bereits beliebt sind haben einen dadurch einen Vorteil noch mehr neue Nutzer anzuziehen, also welche die neu auf dem Markt sind.

TODO: Hyoothese ausformulieren

H 8. *Erfolgreiche Projekte werden noch erfolgreicher. => neue Projekte müssen raus stechen sonst haben sie keine Chance gegen die bestehenden OSS Alternativen*

Ideen für das Erfassen der Daten

Das kann erfassen mittels Umfrage. Durch Fragen wie: "Wie wichtig sind Downloads und GitHub Sterne oder ähnliche Metriken bei der Auswahl eines Projektes, im Vergleich zu anderen Metriken."

Oder: Sortieren der wichtigkeiten, zur Auswahl wird dann sowas gegeben wie:

- Anzahl der Contributor
- Regelmäßigkeit der Commits
- Anzahl der offenen Issues (bzw Tickets o.ä.)
- Beliebtheit des Projekts, anhand Downloads oder GitHub Sterne.
- Lizenz

3 Crawler

Mithilfe des Crawlers kann sowohl der Markterfolg als auch der technische Erfolg gemessen werden. Über Metriken wie GitHub Sterne, Forks oder Downloads lässt sich der Markterfolg quantifizieren. Der technische Erfolg findet sich wieder in Daten wie Anzahl an Commits, Frequenz von Commits oder Mitwirkende.

Bei der ersten Recherche wurde festgestellt, dass die Repositories mit den meisten Github Sternen keine Software direkt sind, sondern Repositories mit Lerninhalten. Zum Zeitpunkt des Schreibens (13. Februar 2022) werden Platz eins, drei und vier von

- freeCodeCamp *(340k Sterne)*,
- free-programming-books *(222k Sterne)* und
- coding-interview-university *(209k Sterne)*

belegt.

Insgesamt sind unter den Top 10 Repositories auf GitHub nur drei IT Projekte im eigentlichen Sinne dabei [Top].

- Vue *(193k Sterne)*,
- React *(182k Sterne)* und
- TensorFlow *(163k Sterne)*.

Es wird daher nicht möglich sein die Top 100 Projekte für die Datensammlung zu verwenden. Stattdessen werden Projekte von Hand ausgewählt, um zu garantieren, dass es sich nur um IT Projekte handelt.

3.0.1 Der Crawler

Hilfreiche Ressourcen:

- Welche Daten könnte man erfassen und wie? Sieh npms.io.
- Hilfreiches Tool: npmtrends.com
- API um an NPM Downloads ran zu kommen: GitHub registry bzw. Deprecated registry
- Resolution Time + Open Percentage auf isitmaintained.com

3.0.2 Disclaimer

npm downloadzahlen sind manipulierbar, ich geh zwar davon aus, dass es allgemein nicht häufig gemacht wird und da ja sowieso vielmehr in die rechnung rein geht als nur npm downloads. Werden diese daten trotzdem mit erfasst. How to exploit NPM.

In einem Blogpost erklärt NPM wie es zu den Download zahlen zustande kommt.

3.1 GitHub API

GitHub stellt eine API bereit welche man zum Analysieren von Daten verwenden kann, dies wird auch um einiges stabiler funktionieren als ein Web Crawler. Hier sind einige Endpoints die sehr Praktisch sein könnten. Mehr gibt es hier. Es wäre wahrscheinlich, sich einen Access Token zu besorgen, ohne sind 60 Aufrufe pro Stunde möglich, mit Token 5000/h.

Als Alternative zur REST API gibt es auch eine GraphQL API. Diese ist etwas präziser und sollte auch z.B. Stars over time geben können.

- Lizenzen
- Project Health und mehr
- Page Views
- Contributers [1]
- Commits [2]
- Stars

[1] Um an die gesamtzahl der Contributer ran zu kommen ist es ein bisschen tricky, sieh Stackoverflow link ich habe dazu auch nicht den Link in der Doku gefunden aber das wird da schon irgendwo sein. Mehr dazu auch hier

[2] Hier muss man für Gesamt Commits den gleichen Trick anwenden nehme ich an wie bei den Contributorn. zusätzlich dazu lässt sich auch eine Zeitspanne bestimmen!

Zusätzlich oder alternativ lässt sich auch mit den GraphQL Endpoints arbeiten. Hier gibt es zwei Beispiele wie andere es gemacht haben. star-history und vesoft-inc. (Wäre praktische diese wahrscheinlich zu Zitieren, die Idee ist ja nicht so ganz meine I guess so ya know). Wie das erste funktioniert weiß ich nicht genau, aber das zweite Tool verwendet GraphQL. Es konnte eine Grafik für React anzeigen von Anfang bis heute mit Sternen, allerdings hat der Request auch irgendwie idk 10min gedauert, aber bei 187k Sternen und damit dem meisten ist das schon durchaus verkraftbar. Die Commits lagen nur ein Jahr zurück, könnte aber eine Implementierungssache sein, da beide Projekte Open Source sind kann man in den Code reinschauen und sich was daraus gönnen. Außerdem gibt es noch folgendes Tool: starline

4 Methodik

Die Datenerhebung findet auf Basis von öffentlichen Daten statt, die man auf den Project-Pages auf GitHub findet. Bei den Downloads handelt es sich, um die npm[1] Downloads.

Die GitHub Daten werden mittels der GitHub API erfasst.

4.1 Umfrage

4.2 Datenerhebung bzw Crawler

Hierfür wurden X viele Projekte Handpicked ausgewählt, von großen mit über hundert tausend GH-Stars, bis hin zu kleinen mit wenigen hundert GH-Stars. Hierbei wurden nur welche ausgewält die....

4.3 Dokumentation

Der Punkt *Gute Dokumentation* (kann/wird) auf zwei verschiedene Arten und Weisen erfasst. Einmal durch die Umfrage und einmal mittels Crawler. In der Umfrage gibt es konkrete Fragen Bezüglich Dokumentation wie:

- Wie wichtig ist eine gute Dokumentation bei der Auswahl einer OSS für Sie?
- Würden Sie wegen einer schlechten Dokumentation ein Stück Software NICHT nutzten?
- Etc.

Bezüglich Datenerfassung mittels Crawler muss man das wohl eher händisch machen. Da man mit einem Crawler nicht bewerten kann, ob eine Dokumentation gut/vorhanden ist vor allem, weil sich diese meist auf anderen Websites befinden. Hierbei könnte man pro Projekt folgendermaßen vorgehen: Man geht händisch durch die Projekte durch und bewertet diese nach folgendem Schema

- 0 = keine Dokumentation
- 1 = Basis Dokumentation in der GitHub README (Nur ein Getting Started)
- 2 = Ausführliche Dokumentation (Eigene Website / Code Beispiele etc.)

1 npmjs.com

Literaturverzeichnis

[Ban13] W. Bangerth und T. Heister. What Makes Computational Open Source Software Libraries Successful? *Computational Science & Discovery*, 6(1):015010, Nov. 2013.

[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.

[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.

[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.

[Top] Top GitHub Projects by Stars. `https://github.com/search?l=&o=desc&q=stars%3A%3E100+stars%3A%3E1&s=stars&type=Repositories`.