Masterarbeit

Neue Akteure in der Schweizer Open-Source- Software Bewegung

Wieso und wie neue Akteure Open-Source-Software veröffentlichen

eingereicht an der
Wirtschafts- und Sozialwissenschaftlichen Fakultät
der Universität Bern

Institut für Wirtschaftsinformatik
Abteilung Information Engineering / Information Management

Dr. Matthias Stürmer

eingereicht von
Vidhushan Asokan
von Worb, BE
im 6. Semester
Matrikelnummer: 10-103-349

Studienadresse
Weiermattweg 1
3098 Köniz
v.asokan@students.unibe.ch
Bern, 06.06.2019

Zusammenfassung

Die Veröffentlichung von Open-Source-Software (OSS) wurde zu einem wichtigen strategischen Bestandteil innerhalb von Unternehmen der IT-Industrie. Unternehmen aus anderen Branchen oder Behörden sind hingegen mehrheitlich passive Nutzer von OSS. Die Veröffentlichung von OSS durch Unternehmen aus anderen Branchen oder Behörden ist eine neue Erscheinung. In dieser Arbeit wird gezeigt, warum neue Akteure in der Schweiz OSS veröffentlichen, welche strategische Bedeutung OSS für sie hat und wie die Zusammenarbeit mit externen Akteuren aussieht. Für diese Masterarbeit wurden sechs verschiedene Fälle analysiert. Dafür wurden Interviews mit acht Mitarbeitern von sechs verschiedenen Unternehmen und Behörden aus der Schweiz durchgeführt. Insgesamt führte dies zu rund sechs Stunden Gesprächsmaterial. Die Resultate basieren auf diesen Gesprächen und der wissenschaftlichen Literatur. Es zeigen sich verschiedene Gründe warum neue Akteure in der Schweiz OSS veröffentlichen. Wichtig sind namentlich der Reputationsgewinn, das Networking, die aktive Steuerung in der Entwicklung, die Verbreitung eigener Lösungen und das Sparpotential.

Summary

The release of open source software (OSS) has become an important strategic component within companies in the IT industry. Companies from other industries or authorities, on the other hand, are mostly passive users of OSS. The publication of OSS by companies from other industries or authorities is a new phenomenon. This paper shows why new players in Switzerland publish OSS, what strategic importance OSS has for them and what cooperation with external players looks like. Six different cases were analysed for this master thesis. To this end, interviews were conducted with eight employees of six different companies and authorities from Switzerland. In total, this led to around six hours of discussion material. The results are based on these interviews and the scientific literature. There are various reasons why new players in Switzerland publish OSS. Important factors are, in particular, the gain in reputation, networking, active control in development, the dissemination of own solutions and savings potential.

Danksagung

An dieser Stelle möchte ich mich bei all denjenigen bedanken, die mich während der Anfertigung dieser Masterarbeit unterstützt und motiviert haben.

Zuerst gebührt mein Dank Herrn Dr. Matthias Stürmer, der meine Masterarbeit betreut und begutachtet hat. Für die hilfreichen Anregungen und die konstruktive Kritik bei der Erstellung dieser Arbeit möchte ich mich herzlich bedanken.

Ein besonderer Dank gilt an Markus Tiede von der Baloise Groupe, Timo Grossenbacher von SRF, Michael Stolz und Daniel Zuck von der Six-Group, Henning Henkel und Tobias Denzler von der Helvetia Versicherung, Thomas Joos vom Amt für Informatik und Organisation des Kantons Bern und David Ösch vom Bundesamt für Landestopografie swisstopo. Ohne die Interviewpartner hätte diese Arbeit nicht entstehen können. Mein Dank gilt ihrer wertvollen Zeit, ihrer Informationsbereitschaft und ihren interessanten Beiträgen und Antworten auf meine Fragen.

Ebenfalls möchte ich mich bei meinen Freunden Jonas, Omid, Artem, Phillipp, Shin, Jan, Armon, Lucca und Klara, bedanken, die mir mit viel Geduld, Interesse und Hilfsbereitschaft zur Seite standen. Bedanken möchte ich mich für die zahlreichen interessanten Debatten und Ideen, die maßgeblich dazu beigetragen haben, dass diese Masterarbeit in dieser Form vorliegt.

Außerdem möchte ich Lucca und Klara für das Korrekturlesen meiner Masterarbeit danken.

Abschließend möchte ich mich bei meinen Eltern und meiner Schwester bedanken, die mir mein Studium durch ihre Unterstützung ermöglicht haben und stets ein offenes Ohr für mich hatten.

Vidhushan Asokan

Bern, 06.06.2019

Gender-Erklärung

Aus Gründen der besseren Lesbarkeit wird in dieser Masterarbeit die Sprachform des generischen Maskulinums angewandt. Es wird an dieser Stelle darauf hingewiesen, dass die ausschließliche Verwendung der männlichen Form geschlechtsunabhängig verstanden werden soll.

Inhaltsverzeichnis

1 Steigende Relevanz von Open Source Software

Der Begriff Open Source Software (OSS) war lange Zeit nur in der Informatikbranche bekannt (Osterloh, Rota, & Kuster, 2004, S.1). Der Prozess der Open Source Softwareentwicklung (OSSE) entspricht nicht dem, was die meisten Ökonomen erwarten würden. Private Unternehmen versuchen normalerweise, die durch sie geschaffenen Werke zu kontrollieren und zu schützen. Insbesondere gilt das für die Rechte an Programmcodes. Demgegenüber wird in OSS-Projekten urheberrechtlich geschütztes Material unter bestimmten Bedingungen für andere zur Nutzung öffentlich zugänglich gemacht. In vielen Fällen muss sich jeder, der das Material nutzt, damit einverstanden erklären, alle Verbesserungen am Originalmaterial unter den gleichen Bedingungen zur Verfügung zu stellen (Lerner & Tirole, 2005, S.99).

In seinen Anfängen war quelloffene Softwareentwicklung eine Nischenerscheinung und vom allgemeinen Markt abgekoppelt. Heutzutage sind OSS-Projekte ein wichtiger Bestandteil der Innovationsstrategien von grossen IT-Unternehmen (Schrape, 2016, S.9). Bei einigen grossen IT-Unternehmen ist OSS schon länger ein Teil der Unternehmensstrategie. IBM soll beispielsweise allein im Jahr 2001 über 1 Milliarde Dollar für OSS-Projekte ausgegeben haben (Lerner & Tirole, 2005, S.99).

Microsoft hingegen sah OSS, insbesondere dem Linux Betriebssystem lange Zeit als deren Hauptkonkurrenten (Dernbach, 2002). In einem Interview mit der Chicago Sun-Times sagte der damalige Microsoft-CEO Steve Ballmer: *«Linux is a cancer that attaches itself in an intellectual property sense to everything it touches»* (Greene, 2001). Aber auch bei Microsoft hat sich die Einstellung gegenüber OSS innerhalb der letzten Jahre drastisch verändert. Im Juni 2018 kaufte Microsoft die Softwareentwicklungs-Plattform GitHub für 7,5 Milliarden US-Dollar und beim heutigen Microsoft-CEO Satya Nadella stehen die Entwickler im Fokus (Perler, 2018). In einem Blog-Beitrag schrieb Nadella zum Kauf von GitHub: “Microsoft is a developer-first company, and by joining forces with GitHub we strengthen our commitment to developer freedom, openness and innovation” (Microsoft News Center, 2018).

Nicht nur bei den grossen Software-Unternehmen, sondern auch in der Schweiz ist OSS aktueller denn je. In einer 2018 durchgeführten OSS-Studie mit 243 Unternehmen und Behörden, geben 60% der befragten an, dass die Bedeutung von OSS zugenommen hat. Die Ergebnisse der Studie zeigen weiter auf, dass die Nutzung von OSS in der Schweiz mit durchschnittlich 7.2 Prozentpunkte Wachstum gegenüber 2015 in praktisch allen Anwendungsgebieten weiter zugenommen hat (Stürmer & Gauch, 2018).

Seit kurzem nutzen Unternehmen und Behörden in der Schweiz nicht nur OSS, sondern es finden auch erste Veröffentlichungen von Codes oder ganzen Projekte von Unternehmen und Behörden, welche nicht aus der IT Industrie stammen, statt. So nimmt zum Beispiel der Kanton Bern eine Pionierrolle ein, indem er im Herbst 2018 eine eigene Software unter einer OS-Lizenz veröffentlicht hat (Zellweger & Preisig, 2018). Aber auch in anderen Bereichen und Branchen wie zum Beispiel im Journalismus, beim Bundesamt für Landestopografie oder bei Versicherungen befasst man en sich mit dem Thema der Veröffentlichung von Codes oder Projekten (Baloise, 2019; SRF Data, 2019a; Swiss Geoportal, 2019)

Weil dies eine neue Erscheinung ist, besteht eine Forschungslücke bezüglich des Nutzens und der Risiken von nicht IT-Unternehmen oder Behörden in der Schweiz, die einzelne Codes oder ganze Projekte unter einer OS-Lizenz veröffentlichen. Dabei drängt sich die Frage nach dem Auslöser für die Veröffentlichung auf. Zudem wäre es spannend zu erfahren, wie nicht IT Unternehmen und Behörden intern aufgestellt sind und eigene Quellcodes veröffentlichen. Zum Schluss wäre es noch interessant zu erfahren, wie die neuen Akteure im Zusammenhang mit OSS mit Externen zusammenarbeiten. Daraus lassen sich vier Forschungsfrage ableiten, welche mit der vorliegenden Arbeit beantwortet werden sollen:

1. Was treibt die neuen Akteure an, eigene Codes oder Projekte unter einer OS-Lizenz zu veröffentlichen?
2. Welchen Nutzen und welche Risiken sehen die neuen Akteure, wenn sie eigene Codes oder Projekte unter einer OS-Lizenz veröffentlichen?
3. Wie wird OSS bei den neuen Akteuren adaptiert und welche strategische Bedeutung hat OSS für sie?
4. Wie arbeiten die neuen Akteure mit Externen zusammen?

Das Ziel dieser Arbeit ist, anhand theoretischer als auch empirischer Erkenntnisse die Forschungslücke zu schliessen, indem versucht wird zu verstehen, warum kommerzielle nicht IT-Unternehmen und Behörden in der Schweiz einzelne Codes oder ganze Projekte unter einer OS-Lizenz veröffentlichen. Dabei sollen Treiber, Nutzen und Risiken erforscht werden. Des Weiteren soll die strategische Bedeutung eruiert werden, indem versucht wird herauszufinden wie neue Akteure in der Schweiz OSS adaptiert haben, wie sie intern im Zusammenhang mit OSS aufgestellt sind und wie sie bei der Veröffentlichung von OSS vorgehen. Abschliessend soll die Frage beantwortet werden, wie nicht IT Unternehmen und Behörden in der Schweiz mit Externen zusammenarbeiten. Die theoretische Untersuchung umfasst das Studium der entsprechenden wissenschaftlichen Literatur. Der empirische Teil setzt sich aus der Analyse von sechs Fallstudien und entsprechenden Experteninterviews mit Mitarbeiter aus nicht IT-Unternehmen und Behörden aus der Schweiz zusammen.

2 Literaturreview

Im folgenden Abschnitt wird als erstes auf die Geschichte der OSS-Bewegung eingegangen. Danach wird ein Überblick über die bestehende Forschung zu folgenden Themen gegeben: die Gründe, wieso IT-Unternehmen OSS veröffentlichen, wie OSS durch die Unternehmen adaptiert wird und was bei der Entwicklung von OSS-Strategien zu beachten ist. Zum Schluss wird auf die Veröffentlichung von OSS durch die öffentliche Hand und die Zusammenarbeit mit Communities eingegangen.

2.1 Wie OSS für Unternehmen tauglich wurde

Um zu verstehen, wieso neue Akteure bei OSS-Projekten beitragen und eigene Projekte unter einer OS-Lizenz veröffentlichen, wird als erstes die Geschichte der OSS-Bewegung kurz zusammengefasst. Wie aus Abbildung 1 hervorgeht, lässt sich die Open-Source-Softwareentwicklung in drei Phasen aufteilen. In einem Beitrag von Schrape (2017) wird aufgezeigt, wie die Utopie «Free Software» von Richard Stallman, zu Open Source als Methode und schlussendlich zu einer Innovationsstrategie von IT Unternehmen wurde.

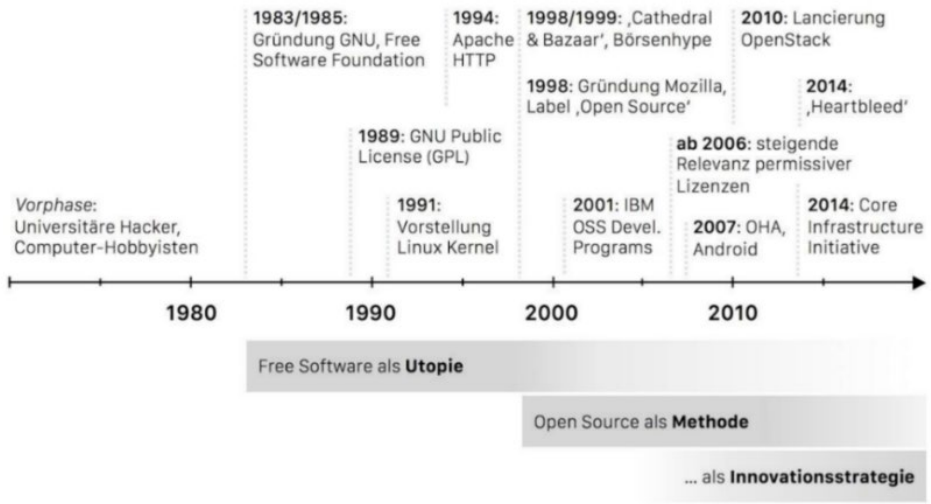

Abbildung 1:Phasen der Open-Source-Softwareentwicklung nach Schrape (2017, S.2)

2.1.1 Free Software als Utopie

Bis in die 1960er waren die Nutzer von Computerhardware tendenziell Universitätsforscher und kommerzielle Forschungs- und Entwicklungseinheiten, die Software als ein von allen Nutzern zu entwickelndes und verbesserndes Forschungsinstrument betrachteten (Gulley & Lakhani, 2010, S.6). Die Software kann

entweder im Quellcode oder im Objekt- bzw. Binärcode übertragen werden. Der Quellcode ist der Code, welche Programmiersprachen wie Basic, C und Java verwendet. Der Objekt- oder Binärcode ist die Folge von 0 und 1, die direkt mit dem Computer kommuniziert, die aber für Programmierer schwer zu interpretieren oder zu ändern ist. In den Anfangszeiten war die gemeinsame Nutzung des Quellcodes für Computerbetriebssysteme und für weit verbreitete Übertragungsprotokolle durch Programmierer in verschiedenen Organisationen üblich. Es wurde auf einer informellen Basis zusammengearbeitet und in der Regel wurden keine Anstrengungen unternommen, um Schutzrechte abzugrenzen oder die Wiederverwendung der Software einzuschränken (Lerner & Tirole, 2005, S.101).

Zu Beginn der 1980er erwies sich die mangelnde rechtliche Absicherung als Problem, weil die Software jedem offen stand und kaum von Einzelaneignungen geschützt war (Schrape, 2017, S.2). Als es kartellrechtlich möglich war, wurde 1983 die Betriebssystemsoftware UNIX, zu der viele Wissenschaftler und Unternehmensforscher anderer Unternehmen beigetragen hatten, von AT&T kommerzialisiert und AT&T begann seine (angeblichen) geistigen Eigentumsrechte durchzusetzen (Lerner & Tirole, 2005, S.101; Schrape, 2017, S.2). Ein weiteres Problem für Unternehmen war, dass Software in diesem Umfeld gerne weitergegeben, aber selten dafür bezahlt wurde. Daraufhin wurden Softwareprodukte ohne den lesbaren Quellcode, sondern nur noch als Binärdatein vertrieben (Schrape, 2017, S.2f.).

Als Reaktion auf die Kommodifizierung von Software gründete der MIT-Mitarbeiter Richard Stallman im Jahr 1985 die Free Software Foundation mit dem Ziel, eine breite Palette von Software kostenlos zu entwickeln und zu verbreiten. Hierzu führte die Free Software Foundation 1989 ein formales Lizenzierungsverfahren, eine so genannte General Public License (kurz GPL), für ein Computer-Betriebssystem namens GNU ein. Der Name GNU ist ein rekursives Akronym, das für "GNU's Not UNIX" steht. (Lerner & Tirole, 2005; S.101; Schrape, 2017, S.2). Die Philosophie der Stiftung lautet, *«dass Nutzer die Freiheit haben Software auszuführen, zu kopieren, zu verbreiten, zu untersuchen, zu ändern und zu verbessern»* (gnu.org, 2019). Die Lizenz zielte darauf ab, dass kooperativ entwickelte Software nicht den Urheber- oder Patentrechten unterliegt. Um die GNU-Software zu modifizieren und zu vertreiben, mussten Softwareentwickler im Gegenzug den

Quellcode frei zur Verfügung stellen und darauf bestehen, dass andere, die den Quellcode verwenden, dies ebenfalls tun. Darüber hinaus mussten alle Erweiterungen des Codes zu den gleichen Bedingungen lizenziert werden. Diese Art von Lizenz wird manchmal als «starkes Copyleft» bezeichnet, denn wenn das Urheberrecht (Copyright) versucht, geistiges Eigentum privat zu halten, versucht Copyleft, geistiges Eigentum frei und verfügbar zu halten (Lerner & Tirole, 2005, S.101f.). Der Erfolg des GNU-Projekts hielt sich in Grenzen, weil das GNU Betriebssystem auf kostenintensiven Workstations zugeschnitten war und unter der ideologisch geprägten Lizenz GPL lief. Trotzdem zeichnete sich der Einfluss kommerzielle Unternehmen bereits in dieser Phase ab. Obwohl die Free Software Foundation als Reaktion auf die Kommodifizierung entstand, konnten 1988 erste kommerzielle Grossspender wie Sony oder Hewlett-Packard angeworben werden (Schrape, 2017, S.3).

Die weite Verbreitung des Internetzugangs in den frühen 90er Jahren führte dazu, dass das Beitragsvolumen und die Vielfalt der Beitragenden stark anstiegen und es entstanden zahlreiche neue quelloffene Projekte, weil der Zugriff auf Projekte und die Koordination erleichtert wurde (Lerner & Tirole, 2005; S.102; Schrape, 2017, S.3) Ein wichtiges Projekt war das Linux-Kernel-Projekt, welches 1991 durch den Studenten Linus Torvalds als freier Betriebssystemkern für die günstigeren Mikrocomputer vorgestellt wurde und deshalb für eine größere Zahl an Entwickler attraktiv war. Trotzdem war Linux am Anfang auch nur in Expertenkreisen bekannt (Schrape, 2017, S.3).

2.1.2 Open Source als Methode

Der Beitrag *«The Cathedral and the Bazaar»* vom Softwareentwickler Eric S. Raymond (1999) führte dazu, dass Linux auch in der allgemeinen Öffentlichkeit wahrgenommen wurde. Er verglich die klassischen Produktionsmodelle als Kathedrale, welche hierarchisch organisiert sind und nur den Quellcode von vollständigen Versionen publiziert werden. Dagegen verglich er Projekte wie Linux mit einem Basar, in welchen der Quellcode stets einsehbar ist und die Gruppe sich durch eine horizontale Struktur und Selbstorganisation ohne zentrales Management auszeichnet. Zu dieser Zeit haben immer mehr IT-Unternehmen ihre Softwareentwicklung in den quelloffenen Bereich ausgelagert. Das bekannteste Beispiel war das Unternehmen Netscape, welche seinen Webbrowser Netscape Communicator in das quelloffene Projekt Mozilla überführte. Netscape handelte so,

weil es in erster Linie neue Kundenkreise erschliessen wollten, nachdem die Gefahr entstand, dass Microsoft den Netscape Navigator durch den in Windows integrierten Internet Explorer aus dem Markt drängen würde (Schrape, 2017, S.4)

Kurz nach der Ankündigung der Veröffentlichung des Netscape-Quellcodes bildete sich 1998 eine Gruppe um Eric Raymond mit dem Glauben, dass die pragmatischen, geschäftskritischen Gründe, die Netscape motiviert hatten, ihren Code freizugeben, eine wertvolle Möglichkeit darstellten, mit potenziellen Software-Anwendern und Entwicklern in Kontakt zu treten und sie davon zu überzeugen, Quellcodes zu erstellen und zu verbessern, indem sie Teil einer engagierten Community sein können (Opensource.org, 2019a). Die Gruppe hielt es auch für sinnvoll, ein einziges Label zu haben, welches die Überlegenheit des Entwicklungsmodells betont und sich vom ideologisch ausgerichteten Label *«Freie Software»* unterscheidet, weil dieses für die Verbreitung quelloffener Methoden in kommerziellen Kontexten hinderlich sein könnte. So wurde die Open-Source-Initiative gegründet (Opensource.org, 2019b; Schrape, 2017, S.4)Die Open Source Initiative hat eine offizielle Definition des Begriffs Open Source entwickelt, welche zehn Kriterien umfasst. Zu den Kriterien gehören unter anderem die freie Weitergabe der Software, das Erlauben von Weiterentwicklungen und Veränderungen, die Unversehrtheit des Quellcodes des Autors und dass eine Diskriminierung gegen Personen, Gruppen oder das Einsatzfeld der Software nicht erlaubt ist (Opensource.org, 2019b). Die Definition der Open-Source-Initiative unterscheidet sich nicht gross von der oben erwähnten Definition der Free Software Foundation, da beide Begriffe ähnliche Kategorien von Software, jedoch auf grundlegend unterschiedliche Basiswerten basieren. Gemäss Stallman ist Open Source eine Entwicklungsmethodik und freie Software eine soziale Bewegung (Stallman, 2009, S.31).

Die Open Source Initiative anerkennt mittlerweile über 80 verschiedene OS-Lizenzen (Open Source Initiative, 2019c). Eines der Hauptunterscheidungsmerkmale zwischen den verschiedenen OS-Lizenzen ist der Grad der Einschränkungen, die der Fähigkeit der OS-Lizenzen auferlegt werden. Die Mehrheit der OS-Lizenzen kann in drei Typen eingeteilt werden (Sen, Subramaniam, & Nelson, 2008, S-211f.). Neben den zuvor beschriebenen «starken Copyleft-Lizenzen», welche sicherstellen, dass alle Erweiterungen des Codes zu den gleichen Bedingungen lizenziert werden müssen, gibt

es sogenannte «schwache Copyleft-» und «permissive-» Lizenzen. Eine Lizenz wird als eine schwache Copyleft-Lizenz bezeichnet, wenn nicht alle abgeleiteten Komponenten eines Werks die Lizenz erben müssen. Dies ermöglicht die Einbindung in proprietäre Produkte, sofern ebendiese abgeleiteten Komponenten quelloffen bleiben. Ein Beispiel dieser Kategorie ist die GNU Lesser General Public License (LGPL). Die permissiven Lizenzen hingegen ermöglichen es, gespaltene Versionen des Programms neu zu lizenzieren, insbesondere auch unter einer proprietären Lizenz. Ein Beispiel dieser Kategorie ist die Berkeley Software Distribution License (BSD-Lizenz). Die permissive-Lizenz ist am wenigsten einschränkende Kategorie unter den genannten OS-Lizenzarten (Sen et al., 2008, S. 211ff.).

Diese alternativen Ansätzen zur Lizenzierung erweiterte die strategischen Möglichkeiten für kommerzielle Unternehmen(Schrape, 2017, S.4) (Schrape, 2017, S.4). In den letzten neun Jahren lässt sich ein Trend hin zu permissiven Lizenzen feststellen (Goldstein, 2018). Während 2010 noch mehr als die Hälfte der meistgenutzten quelloffenen Softwarelizenzen starke Copyleft-Lizenzen waren, dominieren heute die permissiven Lizenzen. Die Entwicklung wird in Abbildung 2 grafisch illustriert.

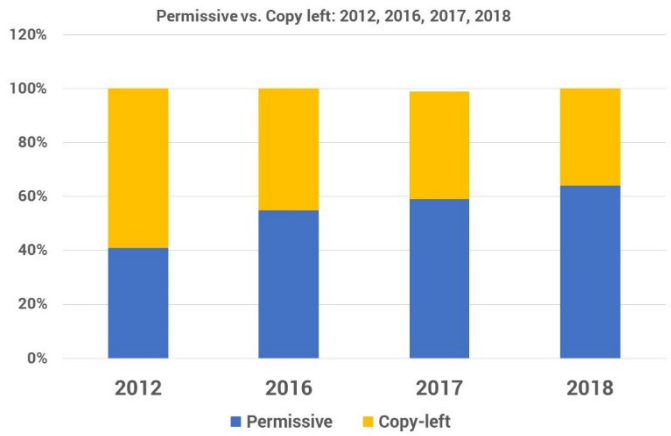

Abbildung 2: Vergleich der permissiven und Copyleft Lizenzen nach Goldstein (2018)

Neben den alternativen Lizenzarten erhielt die OSS-Methode dank Börsengänge von OSS-Unternehmen einen zusätzlichen Aufschwung. So feierte der Linux Distributor Red Hat einer der erfolgreichsten Debüts aller Zeiten an der Börse und erregte eine entsprechend große mediale Aufmerksamkeit (Schrape, 2017, S.4).

2.1.3 Open Source als Innovationsstrategie

Es ist ein stetiger Wandel von OSS als Gemeinschaft einzelner Entwickler zu Open Source Software als Gemeinschaft kommerzieller Organisationen zu beobachten (Ågerfalk & Fitzgerald, 2008, S.404). Während OSS früher nur als Imitationen proprietärer Angebote betrachtet wurden, erkennen Unternehmen heute die OSS-Bewegung als eine wichtige Quelle für Innovationen (Bonaccorsi & Rossi, 2006, S.49; Ebert, 2007, S.105). Dies ist auch daran zu erkennen, dass sich die meisten grossen IT-Unternehmen wie Microsoft, Apple, Google, IBM, Intel, Hewlett-Packard, SAP, Oracle oder Adobe in Form von finanziellen Investitionen und Entwicklungen am Code durch eigene Mitarbeiter an OSS-Projekten beteiligen. Durch dieses Kombination tragen die jeweiligen Unternehmen zu einer Erhöhung der Planungssicherheit in den Communitys bei, sichern sich jedoch auch einen nicht zu unterschätzenden Einfluss auf relevante Entwicklungsvorhaben und Innovationen. Ihre Beteiligung sind nicht ideologisch geprägt, sondern entstehen durch unternehmerisches Kalkül. (Schrape, 2017, S.6f.).

Von Hippel und von Krogh (2003) liefern mit ihrem privat-kollektiven Innovationsmodell eine Erklärung, warum es für Unternehmen Sinn macht ihre Codes und Softwares zu veröffentlichen, auch wenn Wettbewerber darauf Zugriff haben. Das privat-kollektive Innovationsmodell kombiniert zwei verbreitete Innovationsmodelle der Organisationswissenschaft. Einerseits das Modell der privaten Investition, welches davon ausgeht, dass die Innovatoren eine Rendite aus Innovationen erzielen und durch geistige Eigentumsrechte z.B. durch Patente oder Lizenzen einen Anreiz für die Investition schaffen. Jeder Wissensaustausch wird den Nutzen für den Innovator verringern. So ist frei offenbartes Wissen nicht im Interesse des Innovators. Demgegenüber steht das kollektive Innovationsmodell, welches davon ausgeht, dass Innovatoren unter Bedingungen des Marktversagens zusammenarbeiten, um ein öffentliches Gut zu produzieren. In diesem Fall profitieren die Innovatoren nicht mehr als alle anderen, die nicht in das Gemeinwohl investieren, so dass es zu Trittbrettfahrern kommen kann. Das privat-kollektive Innovationsmodell basiert auf der Annahme, dass ein Innovator, der privat ein öffentliches Gut schafft, mehr davon profitieren wird als jemand der nichts dazu beiträgt und das öffentliche Gut nur nutzt. Das Ergebnis der Investition steht allen gleichermassen zur Verfügung, jedoch

profitiert der Innovator des Schaffungsprozesses des öffentlichen Gutes (Hippel & Krogh, 2003).

Bei OSS ist der Quellcode in Form von explizitem Wissen frei für jedermann zugänglich und Unternehmen, welche im Entstehungsprozess mitwirken, gewinnen dadurch an implizitem Wissen und Expertise, welches einen schwer nachahmbaren Vorteil darstellt (Grand, von Krogh, Leonard, & Swap, 2004, S.601). Unternehmen, die auf Innovationen aus der Community setzen, sind zudem proaktiver bei der Freigabe von Code. Dies ist eine praktikable Strategie, weil je früher der Code freigegeben wird, desto eher wird er als Grundlage für neue Innovationen dienen (Dahlander, 2005, S.281f.).

Die Open Innovation, die in der Entwicklung von Open Source Software zu sehen ist, kann als neue Form des Outsourcings von Innovationen gesehen werden. Es gibt jedoch einen entscheidenden Unterschied zwischen Open Innovation und Innovation Outsourcing. Bei Open Innovation geht es nicht darum, die Kosten zu reduzieren, indem Unternehmen den Umfang der intern durchgeführten Forschungs- und Entwicklungsaktivitäten reduzieren. Ziel ist viel mehr, dass das Unternehmen die Gemeinschaft nutzt, um seine Ressourcen für Forschung- und Entwicklung zu erhöhen. Diese Erhöhung der Ressourcen macht es möglich, dass die Produkte schneller und besser entwickelt werden, als wenn alle Ideen und Lösungen im Unternehmen entstünden (Andersen-Gott, Ghinea, & Bygstad, 2012, S.108). Neben der erhöhten Innovationsfähigkeit dank OSS, gibt es noch weitere Gründe für Unternehmen in OSS-Projekten beizutragen und eigene OSS zu veröffentlichen, welche im folgenden Kapitel erläutert werden.

2.2 Wieso Unternehmen sich in OSS-Projekten engagieren

Es gibt eine Vielzahl an Studien, welche die Motive für das Beitragen von freiwilligen Programmierern untersuchten (Osterloh et al., 2004; von Krogh, Haefliger, Spaeth, & Wallin, 2012). Jedoch gibt es erst wenige Studien, die Motive und Nutzen von Unternehmen für das Engagement in OSS- Projekten analysiert haben. In der untersuchten Literatur bezüglich des Nutzens für IT-Unternehmen wurde nicht zwischen dem Beitragen in anderen OSS Projekten und dem veröffentlichen von eigenen OSS-Projekte unterschieden, sondern als Engagement in OSS-Projekten

zusammengefasst (Andersen-Gott et al., 2012; Bonaccorsi & Rossi, 2006; Wichmann, 2002)

2.2.1 Wieso grosse Softwareunternehmen in OSS-Projekten beitragen

Wichmann (2002) untersuchte 2001 die OSS-Aktivitäten von 25 grossen Softwareunternehmen und ging der Frage nach, wieso Unternehmen erhebliche Ressourcen in Open-Source-Aktivitäten investieren, um der Öffentlichkeit Software zur Verfügung zu stellen. Zusammenfassend lässt sich feststellen, dass fast ein Drittel der 25 größten Softwareunternehmen im Jahre 2001 (32%) an großen OSS-Entwicklungsaktivitäten beteiligt waren. 12% (drei Unternehmen) hatten kleinere Projekte und die Mehrheit (56%) hatte keine sichtbaren Open-Source-Projekte. In diesen Unternehmen können jedoch Einzelpersonen an OSS-Aktivitäten beteiligt gewesen sein. In der Analyse kristallisierten sich vier Schlüsselmotive heraus, warum sich grosse Unternehmen an der Entwicklung von Open-Source-Software beteiligen. Die Schlüsselmotive sind Standardisierungsgründe, Kostensenkungen, strategische Ziele und Kompatibilitätserzwingung. Gründe für Standardisierung und Kostenersparnisse sind langfristig ausgelegt und beziehen sich in den meisten Fällen auf die Infrastrukturen, wie zum Beispiel die Unterstützung von Linux, damit eigene Softwarekomponenten durch OSS-Komponenten ersetzet werden können. Dagegen sind strategische Ziele einzelne Softwareprodukte als OSS zur Verfügung zu stellen, um die eigene Software mit OSS kompatibel zu machen, tendenziell kurzfristig ausgelegt und konzentrieren sich auf einzelne Softwareprodukte.

2.2.2 Steigende moralische Verpflichtung von Unternehmen

Bonaccorsi und Rossi (2006) verglichen die Motivationen einzelner Programmierer und kommerzieller Unternehmen. Dabei unterteilen sie die Motive in wirtschaftliche, soziale und technologische (vgl. Tabelle 1). Es wurden signifikante Unterschiede zwischen den Motivationen von Einzelpersonen und denen von Unternehmen gefunden. Insbesondere betonen die Unternehmen wirtschaftliche und technologische Gründe für den Einstieg und den Beitrag zu OSS. Soziale Motive haben bei den Unternehmen eine geringere Bedeutung, die dagegen typisch für einzelne Programmierer sind. Die pragmatischeren Motivationsprofile von Unternehmen (wirtschaftlich & technologisch) werden in der Gemeinschaft von den Programmierern akzeptiert, vorausgesetzt, die Unternehmen halten sich an die Regeln

der Gemeinschaft. Geschäftsmotive zerstören anscheinend nicht die Gemeinschaft, sondern stärken diese sogar.

Wirtschaftlich	• Unabhängigkeit von der Preis- und Lizenzpolitik der großen Softwareunternehmen • Auseinandersetzung mit dem neuen Modell der Software als verbraucherorientierter Dienst (Geld verdienen mit ergänzenden Diensten) • Erzielung indirekter Einnahmen durch den Verkauf verwandter Produkte • Innovationsförderung (durch Nutzung der FuE-Aktivitäten der Open-Source-Community) • Einstellung von guten IT-Spezialisten
Sozial	• Konformität mit den Werten der Open-Source-Community (um das Vertrauen der Open-Source-Entwickler nicht zu missbrauchen) • Austausch von Code und Wissen mit der Gemeinschaft (wechselseitig zur Unterstützung der Zusammenarbeit) • Software sollte kein proprietäres Gut sein (um die Marktmacht der großen Softwareunternehmen zu verringern)
Technologisch	• Nutzung von Rückmeldungen und Beiträgen von Entwicklern der Open-Source-Community zur Senkung der Entwicklungskosten und zur Verbesserung der Software. • Nutzung von Rückmeldungen und Beiträgen der Anwendergemeinschaft zum Testen und Verbessern von Software • Senkung der Hardwarekosten • Förderung der Normung • Behebung von Sicherheitsproblemen

Tabelle 1: Motivationen von Unternehmen zur Teilnahme an OSS-Projekten nach (Bonaccorsi und Rossi (2006, S.48)

In einer Fallstudie von Andersen-Gott et al. (2012) wurden drei verschiedene Unternehmen aus der IT-Dienstleistungsbranche nach deren Treiber befragt. Die drei Haupttreiber für den Beitrag zu OSS sind der Verkauf komplementärer Dienste, Aufbau einer größeren Innovationsfähigkeit und die Kostensenkung durch die Zusammenarbeit mit einer externen Community. Die drei vorgestellten Studien haben Unternehmen aus der IT-Branche befragt oder untersucht. Während Bonaccorsi und Rossi (2006) und Wichmann (2002) die soziale Motive bei Unternehmen eine untergeordnete Rolle spielten, deuten die Ergebnisse in der Fallstudie von Andersen Andersen-Gott et al. (2012) darauf hin, dass es eine Verschiebung in der Sichtweise kommerzieller Unternehmen auf OSS geben könnte, denn die befragten Unternehmen haben sich alle moralisch verpflichtet, einen Beitrag zu OSS zu leisten. Eine Erklärung ist, dass das OSS-Entwicklungsmodell seit der ersten zwei Studien gereift ist und dass

Unternehmen beginnen, ein Gefühl dafür zu entwickeln, wie OSS-Entwicklung vom gegenseitigen Beitrag abhängt, und dass eine solche moralische Verpflichtung innerhalb der Unternehmen und nicht nur auf individueller Ebene zu wachsen beginnt.

2.3 Wie OSS durch Unternehmen adaptiert wird

Die ersten zwei Kapitel haben aufgezeigt, dass OSS ihre Formatierung als Gegenentwurf zur kommerziellen Herstellung proprietären Software in den letzten zwei Jahrzehnten weitgehend verloren hat, da quelloffene Entwicklung sich als Methode und als Innovationsstrategie in der Softwarebranche etabliert hat (Schrape, 2016, S.28). In diesem Kapitel wird beschrieben wie sich Unternehmen und Organisationen im Zusammenhang mit Open Source Software angenähert haben.

2.3.1 Wie Unternehmen mit OSS-Projekten interagieren

Die Literaturüberprüfung von (Hauge, Ayala, & Conradi, 2010) die Annäherung von softwareintensiven Unternehmen an OSS analysiert und dabei in 112 Arbeiten empirische Beweise gefunden, wie diese mit OSS tatsächlich interagieren. In ihrer Analyse haben sie sechs verschiedene Möglichkeiten identifiziert, wie softwareintensive Unternehmen OSS adaptieren (Hauge et al., 2010, S.1136).

1. Einsatz von OSS-Produkten in ihrer Umgebung als Endbenutzer (z.B. Einsatz von Linux)
2. Einsatz von OSS CASE Tools in der Softwareentwicklung (z.B. mit Eclipse)
3. Integration von OSS-Komponenten in das eigene Softwaresystem (z.B. Integration von Google Web Toolkit)
4. Teilnahme an der Entwicklung von OSS-Produkten, die von einer anderen Organisation oder Gemeinschaft kontrolliert werden (z.B. Bei Linux beitragen).
5. Bereitstellung eigener OSS-Produkte und Bezug zu einer Community rund um diese Produkte (z.B. Bereitstellung von MySQL)
6. Verwendung von Softwareentwicklungspraktiken innerhalb eines Unternehmens oder eines Konsortiums von Unternehmen, die oft mit OSS-Communities in Verbindung gebracht werden, (z.B. durch Praktiken wie Code-Sharing, Peer-Review-Verfahren, Benutzerbeitrag)

2.3.2 Sechsstufige Maturitätsmodell

Ein ähnliches Model ist das sechsstufige Maturitätsmodell, welches die Dynamik zwischen Unternehmen und OSS-Projekten erklärt ist in Abbildung 3 zu sehen. Das Maturitätsmodell unterteilt die Möglichkeiten wie Unternehmen mit OSS-Projekten interagieren in sechs Stufen. Im Gegensatz zu der Analyse von (Hauge et al., 2010), welche sich auf die Softwareentwicklung fokussierte, wird im Maturitätsmodell insbesondere die Wertschöpfung, welche durch OSS generiert wird, berücksichtigt (Carbone, 2007).

Die Stufe null ist die Verleugnung, in dem Unternehmen Open Source keinen Wert attestieren oder es nicht benutzen. Die erste Stufe beinhaltet die passive Nutzung von OSS. OSS wird genutzt, um die Entwicklungszeit und -kosten zu reduzieren und bietet keine oder eine geringe Wertschöpfung.

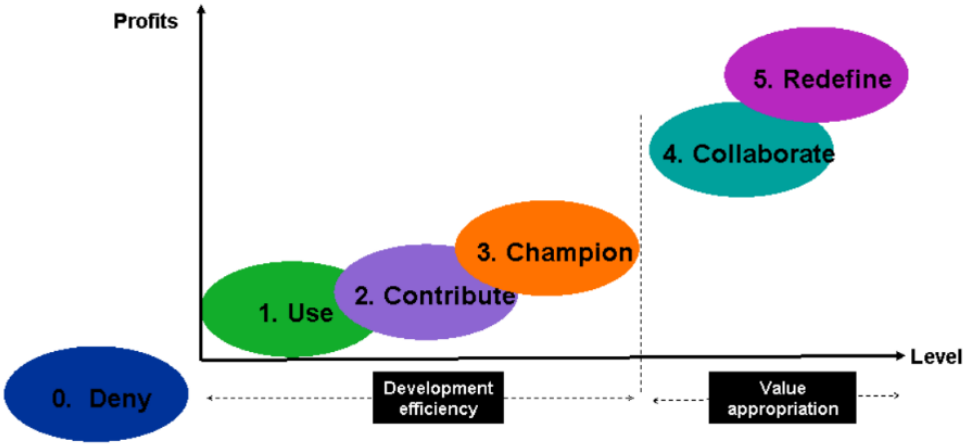

Abbildung 3: Sechsstufige Maturitätsmodell nach Carbone (2007)

Die zweite Stufe wird erreicht, wenn das Unternehmen vom passiven Benutzer zum aktiven Beitragenden wird. In dieser Phase wird mit der Community zusammengearbeitet, indem das Unternehmen Codes einbringt oder sich finanziell beteiligt. Die Community profitiert davon und es verbessert die Softwareverteilung. Der Geschäftsfokus ist relativ begrenzt und es handelt sich lediglich um eine lose Zusammenarbeit zwischen Unternehmen und der Community.

Auf der dritten Stufe wählt und verwaltet das Unternehmen proaktiv eine OSS-Ökosystem, um die Geschäftsziele zu erreichen. Das Unternehmen übernimmt Verantwortung, indem es die Projekte leitet und veröffentlicht. Sowohl die Community als auch das Unternehmen profitieren davon. Die bisher genannten Phasen

sind primär von Entwicklern getrieben und der Fokus liegt in der Entwicklungseffizienz.

Bei den letzten beiden Stufen liegt der Fokus in der Wertschöpfung, welche dank OSS-Projekten generiert werden kann. Auf der vierten Stufe geht es um die Definition eines strategisches Geschäftsmodells, welches auf OSS basiert. Das Unternehmen veröffentlicht aktiv Quellcodes, setzt Kapital ein und übt Einfluss aus, um ihre Geschäftsziele zu erreichen. Durch die Veränderung des Umfelds verschafft sich das Unternehmen einen Wettbewerbsvorteil. Die Community profitiert von den Ressourcen des Unternehmens und kann neue Versionen von Software entwickeln. Die letzte Stufe ist die aggressivste Phase, in der Führungskräfte in Programme und Tools investieren, um Produkte zu entwickeln, die auf OSS basieren. Das Unternehmen gewinnt einen bedeutenden Wettbewerbsvorteil, indem es Veränderungen in mehreren Ökosystemen berücksichtigt. In dieser Phase kommt es auch vor, dass das Unternehmen seine Position in der Wertschöpfungskette ändert. Zum Beispiel kann ein Unternehmen von einem Produkt-Geschäftsmodell zu einem Service-Geschäftsmodell wechseln. Die Community profitiert, da sie projektübergreifende Verbindungen nutzen kann (Carbone, 2007).

Auch Eclipse Foundation verwendet dieses Maturitätsmodell und hat 2009 eine Umfrage mit 1365 Personen gemacht, welche grösstenteils in der IT-Industrie arbeiten. Die Ergebnisse wurden mit einer ähnlichen Umfrage von 2007 verglichen (Eclipse Foundation, 2009, S.5ff.). Praktisch alle Unternehmen befinden sich zumindest auf der Stufe eins, weil sie nutzen OSS. Der Anteil, welche OSS nutzen dürfen, aber vom Unternehmen nicht erlaubt war etwas zurückgeben, sank von 45% auf 27%. Hingegen stieg der Anteil für die Beteiligung in OSS-Communities von 37% auf 48%. Die Ergebnisse zeigen auch, dass immer wie mehr Unternehmen ein Geschäftsmodell mit OSS aufbauen, jedoch war das Wachstum nicht so gross, wie beim Beitragen zu OSS. Die Umfrage illustriert, dass Unternehmen aus der IT-Industrie schon vor zehn Jahren immer offener für die Teilnahme an OSS-Projekten wurden. Die vorgestellten Modelle beschreiben wie Unternehmen in der Vergangenheit OSS adaptierten und welche Möglichkeiten der Veröffentlichung eigener Codes oder Projekten bestehen. Nicht ersichtlich aus den Modellen ist jedoch, wie Unternehmen genau vorgehen müssen, um eine erfolgreiche Open Source Strategie entwickeln zu können.

2.4 OSS Strategieentwicklung in Unternehmen

In diesem Kapitel werden als erstes die Einflussfaktoren, Herausforderung und Verantwortungen bei der Strategieentwicklung in Unternehmen vorgestellt. Danach werden verschiedene Arten der Veröffentlichung von OSS durch Unternehmen erläutert. Zum Schluss wird auf Richtlinien und Compliance eingegangen.

2.4.1 Einflussfaktoren bei der Strategieentwicklung in Unternehmen

Nach (Glynn, Fitzgerald, & Exton, 2005) ist die Adaption von OSS eine komplexe Angelegenheit, welche das externe Umfeld, individuelle Faktoren sowie organisatorischen und technologischen Kontexten beinhaltet. Beim externen Umfeld, wurde zum Beispiel das Bewusstsein, dass andere Organisationen OSS einführen, als wichtig erachtet. Einerseits wird dadurch die Adaption von OSS gefördert, weil Unternehmen es als sicher wahrnehmen, weil andere das Risiko eingehen. Anderseits könnte auch die Befürchtung, dass andere durch OSS einen Wettbewerbsvorteil erzielen könnten, eine Erklärung für die Adaption von Open Source Software sein. Technologisch sind die Veränderbarkeit des Quellcodes und die Verfügbarkeit von Wartung und Support wichtige Faktoren. In Bezug auf organisatorische Kontextfaktoren wird das Budget, Unterstützung durch das Management und kompetentes Personal als wichtig erachtet. Im Hinblick auf die individuellen Faktoren ist es wichtig, dass die OSS-Ideologie von den Mitarbeitern unterstützt wird und ein *«Open Source Software Champion»* existiert (Glynn et al., 2005, S.225ff.).

Zudem muss das Vorhaben von Softwareentwicklern zur Strategie vom Unternehmen passen. Unternehmen müssen sich bewusst sein, dass es nicht nur einen richtigen Weg gibt, OSS einzusetzen. OSS bietet mehrere Möglichkeiten, die jeweils ihre einzigartigen Vor- und Nachteile mitbringen. Jedes Unternehmen sollte die Auswirkungen von Open Source in seinem eigenen Kontext bewerten. Es ist zu beobachten, dass viele Unternehmen Open Source einsetzen, ohne zu wissen, ob sie davon profitieren werden und ohne eine klare Strategie hinter dieser Einführung zu haben (Hauge et al., 2010, S.1145f.).

2.4.2 Mangelnde OSS-Strategien in Unternehmen

Eine mögliche Erklärung für mangelnde Strategie und Verständnis in einem Unternehmen könnte sein, dass der Ursprung einer Grenzüberbrückung meistens bei den Mitarbeiter liegt, indem sie ihr Unternehmen mit externen Informationen

verbinden und es auf diese Weise mit neuen Innovationen in Kontakt bringen. Im Falle von OSS ist es möglich, dass die Adaption hauptsächlich eine *Bottom-up-Initiative* ist, bei der die Mitarbeiter OSS privat nutzen und sie in ihrem Unternehmen einführen wollen. Die Adaption wird mehrheitlich von Ingenieuren als von strategischen Managemententscheidungen getrieben (Hauge et al., 2010; Ven & Verelst, 2006, S.117).

Das *Bottom-Up* Vorgehen steht im Widerspruch zu den meisten herkömmlichen Entwicklungsprozessen (Pizka & Bauer, 2004, S.1). Die Unterstützung des Top-Managements ist zweifellos entscheidend für radikale, risikoreiche Initiativen wie OSS (Glynn et al., 2005, S.227). OSS wird jedoch in vielen Unternehmen immer noch auf technische und rechtliche Belange reduziert, weshalb es zentral ist, dass Unternehmen alle Einflussfaktoren von OSS berücksichtigen und eine Strategie für den Umgang mit OSS vorgeben. Jedoch haben bis heute noch viele Unternehmen keine Strategie (Greve, 2016, S.101).

2.4.3 OSS- Strategieentwicklung ist Aufgabe des Managements

Nach Porter (Porter, 1996) ist die Strategie die Schaffung von einer einzigartigen und wertvollen Position, welche eine Reihe von Aktivitäten beinhaltet. Es erfordert zwischen verschiedenen Möglichkeiten abzuwägen und zu entscheiden, um die Aktivitäten genau aufeinander abstimmen zu können. Strategie und Führung sind untrennbar miteinander verbunden. Damit die Strategie vertieft werden kann, brauchen die Mitarbeiter eine Anleitung.

Die Entwicklung einer geeigneten OSS-Strategie ist nicht die Aufgabe von Entwicklungs- oder Rechtsabteilungen, sondern die Aufgabe des Top-Managements. Für die Strategieentwicklung sollten alle Stakeholder (Entwickler, Rechtsdienst, Marketing, Finanzen etc.) miteinbezogen werden. Dafür müssen zuerst die Randbedingungen und Einflussfaktoren (Markt, Konkurrenten, Kundenstruktur etc.) analysiert werden, welche die Entwicklung der Strategie beeinflussen. OSS kann auch nur ein Mittel sein, um die Geschäftsstrategie umzusetzen. Jedoch braucht es dafür eine klare Strategiedefinition und konsequente Umsetzung. Fragen bezüglich dem Grad der Adaption, der Ziele, der Relevanz für eigenes Produktportfolio, der Ressourcen, des Nutzens, der Risiken, der Risikobereitschaft, der Zusammenarbeit mit OSS- Ökosystem etc. müssen beantwortet werden. Die Strategiedefinition ist sehr marktabhängig, müssen von jedem Unternehmen individuell beantwortet werden und

die genannten Themen sind nicht als abschliessende Liste zu verstehen (Greve, 2016, S.101ff.)

2.4.4 Verschiedene Arten OSS zu veröffentlichen

Es gibt verschiedene Arten OSS zu veröffentlichen. Der kleinstmögliche Beitrag ist, wenn ein Unternehmen oder eine Behörde eine bestehende OSS verwendet und im Anschluss eigene Fehlerbehebungen oder Funktionsverbesserungen freigibt. Wenn sie dies nicht veröffentlichen und nur für sich behalten würden, würde der Fehler im nächsten Release erneut vorhanden sein und neue Funktionen könnten nicht entwickelt werden. Unternehmen und Behörden profitieren von einer höheren Entwicklungsgeschwindigkeit und weniger internen Wiederholungen der Arbeit, wenn sie kleine Verbesserungen stromaufwärts zum Hauptentwicklungszweig der OSS-Lösung beitragen (Free Software Foundation Europe, 2019, S:20).

Der nächste Beitrag wäre, wichtige Weiterentwicklungen von bestehenden OSS-Lösungen mitzufinanzieren oder auch aktiv nach weiteren Sponsoren zu suchen. Oder einen OSS-Dienstleister für die Weiterentwicklung der OSS-Lösung zu beauftragen, statt eigene Projekte zu starten. Durch die gemeinsame Weiterentwicklung von OSS kann die Qualität der Codes verbessert und die Aufwendungen durch die Aufteilung der Kosten gesenkt werden (Free Software Foundation Europe, 2019, S:20).

Der letzte Schritt wäre dann als Behörde oder Unternehmen ein eigenes OSS-Projekt zu starten und den gesamten Quellcode eines Softwareprodukts freizugeben. Dieser Schritt ist eine langfristige Investition, weil für die Vorbereitung und Freigabe des Quellcodes, Koordination mit der Community und eventuell für die Gründung eines unabhängigen gemeinnützigen Vereins Ressourcen benötigt werden, um den Quellcode zu kontrollieren. Wenn der Aufbau von Communities erfolgreich verläuft, wird die Software von anderen weiterentwickelt, was zu einer umfassenderen Lösung führt und die Entwicklungskosten langfristig senkt. Durch die Schaffung einer großen Benutzerbasis wird zusätzlich der Markt für OSS-Dienstleistungen erschlossen. Anbieter wachsen, wodurch die Abhängigkeiten von externen Anbietern verringert werden (Free Software Foundation Europe, 2019, S:20).

2.4.5 Erstellung und Freigabe von OSS

Bei OSS geht man im Allgemeinen davon aus, dass die Software öffentlich erstellt und bereitgestellt wird. Der Begriff OSS sagt genau betrachtet nichts über die Art und Weise aus, wie die Software erstellt wird. Es gibt drei verschiedene Varianten, wie OSS erstellt und bereitgestellt werden kann. Die erste Möglichkeit ist, dass die Erstellung öffentlich erfolgt und für die Allgemeinheit bereitgestellt wird. Die zweite Variante ist, dass die Erstellung nicht öffentlich erfolgt und für die Allgemeinheit bereitgestellt wird. Drittens kann die Erstellung nicht öffentlich erfolgen und nur für bestimmte Nutzer bereitgestellt werden. Für die Geschäftsstrategie ist es entscheidend, dass man den Zweck bestimmt, den man mit der entsprechenden Strategie erreichen will (Greve, 2016, S.106). Unter öffentlicher Erstellung versteht man nicht, dass von Grund auf im Basarstil kodiert werden. Man kann im Basarstil testen, Fehler beheben und verbessern, aber es ist sehr schwierig, ein Projekt ganz öffentlich zu starten. Die Community muss etwas lauf -und testfähiges zum Spielen haben (Raymond, 1999, S.37).

Eine weitere Unternehmensstrategie kann sein, dass die Erstellung nicht öffentlich erfolgt, indem eine bislang proprietäre Software unter einer OS-Lizenz veröffentlicht wird. Zwei Faktoren spielen bei dieser Strategie eine Rolle. Erstens müssen Unternehmen durch die Veröffentlichung einer proprietären Software unter einer OS-Lizenz einen steigenden Gewinn im komplementären Segment erwarten. Zweitens muss der zusätzliche Gewinn den Gewinn, der mit der proprietären Software hätte erzielt werden können, übersteigen. Insbesondere für kleine Unternehmen könnte es sinnvoll sein auf Lizenzverkäufe zu verzichten, den Quellcode zu veröffentlichen und darauf aufbauend komplementäre Produkte und Dienstleistungen anzubieten. Dies ist eine Strategie wie ein kleines Unternehmen im ursprünglichen Segment bestehen könnte, falls es gegenüber einem Marktführer abgeschlagen ist und langfristig nicht überleben kann. Ein passender Vergleich zur Illustration hierzu wäre das Verschenken von Rasierern, um im Anschluss mehr Rasierklingen verkaufen zu können. Im Kontext von Open Source verschenkt man den Code, um einen Nutzen daraus zu ziehen (Lerner & Tirole, 2003, S.225).

2.5 Freigabe von OSS durch die öffentliche Hand

Weltweit haben mehr als einhundert Regierungsbehörden auf nationaler und lokaler Ebene ein Konto auf GitHub für den Austausch und die Zusammenarbeit für OSS-Projekte (GitHub and Government, 2019). Die Liste der staatlich geförderten Softwareprojekte, die ihren Quellcode öffentlich zugänglich machen und mit anderen Institutionen teilen, wird von Tag zu Tag länger. So bietet beispielsweise der Gemeinsamer Bibliotheksverbund (GBV) eine OSS-Lösung an, die von Bibliotheken in ganz Deutschland genutzt wird. Oder der Staat Luxemburg entwickelte ein OSS-System zur elektronischen Patientenakte, welches von vielen Ärzten und Kliniken genutzt wird. Einige OSS-Lösungen wurden sogar international wiederverwendet. Das isländische nationale Geoportal und das nationale Landesvermessungsamt von Moldawien nutzen *«Oskari»*, eine Software zur Visualisierung und Analyse von räumlichen und statistischen Daten, welche von der finnischen Landesvermessung als OSS veröffentlicht wurde (Free Software Foundation Europe, 2019, S:20).

Die Beispiele zeigen auf, dass die Veröffentlichung von eigenem Code oder ganzer Software nicht nur für Unternehmen relevant ist, sondern auch immer wie mehr in der Politik diskutiert wird. Im Herbst 2017 hat die Free Software Foundation Europe (FSFE) die Kampagne *«Public Money? Puplic Code!»* gestartet. Ihre Forderung lautet, dass wenn es sich um öffentliche Gelder handelt, auch der Code öffentlich sein sollte. 165 Organisationen und 21807 Personen unterstützen diesen Aufruf bereits, indem sie einen offenen Brief unterzeichnet haben, welche an Abgeordnete in ganz Europa, die über Softwarefreiheit in öffentlichen Verwaltungen diskutieren, überreicht werden soll (FSFE, 2019).

Im Folgenden wird der Nutzen einer solchen Veröffentlichung durch die öffentliche Hand vorgestellt und anschliessend auf die rechtlichen Grundlagen für die Veröffentlichung von OSS durch die öffentliche Verwaltung in der Schweiz eingegangen.

2.5.1 Nutzen für die öffentliche Hand eigene Software zu veröffentlichen

Eine Strategie der Veröffentlichung von eigenem Code ist auch bei Software der öffentlichen Hand sinnvoll, welche nicht für Kommerzialisierungszwecke, sondern für den Eigengebrauch entwickelt wurde. Unternehmen können durch die Veröffentlichung von Software für den internen Gebrauch einem Wettbewerbsnachteil haben, weil Konkurrenten möglicherweise von internen Informationen, wie

beispielsweise Betriebsabläufe, profitieren könnten. Im Gegensatz zu Unternehmen besteht dieses Risiko bei der öffentlichen Hand nicht, weil der Wettbewerbsdruck gering ist und kaum ein Interesse besteht die Betriebsabläufe geheim zu halten (Poledna, Schlauri, & Schweizer, 2017, S.24f.).

Durch die Veröffentlichung von eigenem Code oder Software profitiert die öffentliche Hand auf verschiedenen Ebenen. Wenn ähnliche Programme nicht komplett neu programmiert werden müssen, können so Steuereinsparungen erzielt werden. Bei grossen Projekten können Expertise und Kosten geteilt werden. Zudem fördern die transparenten Prozesse die Innovation und man muss das Rad nicht neu erfinden (FSFE, 2019). Weiter schafft die öffentliche Hand durch die Veröffentlichung Transparenz. Bürger und Gruppen der Zivilgesellschaft können durch die Veröffentlichung Prozesse besser nachvollziehen, was eine bessere demokratische Kontrolle über die öffentliche Hand erlaubt und zusätzliches Vertrauen schafft. Zusätzlich werden Verbesserungen am Code von der Gemeinschaft erstellt und können von der öffentlichen Hand genutzt werden (Poledna et al., 2017, S.25).

Ein weiteres Argument für den Einsatz und die Publikation von OSS ist die Unabhängigkeit gegenüber Lieferanten (Poledna et al., 2017, S.109). In diesem Zusammenhang wird oft von Vendor Lock-in bzw. Lieferantenbindung gesprochen. Die Lieferantenbindung ist eine Situation, in der ein Kunde, der ein Produkt oder eine Dienstleistung nutzt, nicht ohne Weiteres auf das Produkt oder die Dienstleistung eines Wettbewerbers übergehen kann (WhatIs.com, 2019) . In öffentlichen Verwaltungen gibt es viele Erscheinungsformen der Lieferantenbindung. Beispielsweise Dateiformate von Dokumenten, die nur von einem bestimmten Produkt gelesen werden können, Datenbankinhalte, die nicht in das Format eines konkurrierenden Anbieters konvertiert werden können oder die Einschränkung, ein überteuertes Software-Upgrade zu kaufen, um auf Dateien zugreifen und Sicherheitsfixes erhalten zu können. Mit der Auslagerung von Dienstleistungen und Speicherung der Daten bei Cloud-Anbietern, welche mehr und mehr an Bedeutung gewinnen, wächst das Problem der Lieferantenbindung. Die Kontrolle und das Wissen über die aktuelle Technologie nehmen ab, während die Kosten aufgrund der reduzierten Übersicht leicht explodieren können (Free Software Foundation Europe, 2019, S.8). Durch Kooperation mit einer Entwicklergemeinde für Entwicklung und Support und die mit OSS eingehende Unabhängigkeit von einzelnen Lieferanten, führen zu

Kostenvorteilen auf Seiten der öffentlichen Hand, die den Einsatz von OSS, und insbesondere auch deren Publikation unter einer OSS-Lizenz, rechtfertigen (Poledna et al., 2017, S.109).

Zusätzlich fördert der Einbezug einer Vielzahl von Entwicklern die Produktqualität und -stabilität sowie insbesondere die Produktsicherheit, weil der freie Zugang zum Quellcode nicht nur für die Anpassung und Weiterentwicklung, sondern auch für Prüfzwecke relevant ist. Auch wenn man nicht in der Lage ist, den Code selbst zu überprüfen, führt bereits die freie Verfügbarkeit des Codes zu einem gewissen Vertrauen, weil man davon ausgehen kann, dass die Community dazu in der Lage ist (Poledna et al., 2017, S.18). Durch die Veröffentlichung von OSS werden die Sicherheitsüberprüfungen durch unabhängige Parteien ermöglicht (Free Software Foundation Europe, 2019, S.9). Proprietäre Softwareunternehmen stellen in der Regel ihre eigenen Auditoren ein und die Kunden müssen diesen bei der Sicherheit, der von ihnen verkauften Software, vertrauen. Mit OSS kann jedes Unternehmen oder jede Regierungsstelle ihre eigene Prüfung des Quellcodes oder eines Teils der Anwendung durchführen, an der sie interessiert sind. Sicherheitsprobleme werden öfters und schneller aufgedeckt, indem der Code durch eine relativ grosse Anzahl von Benutzern bearbeitet wird. Voraussetzung eine erhöhten Produktsicherheit von OSS ist das Vorhandensein einer Community, denn ohne Community ist OSS nicht sicherer als proprietäre Software (Free Software Foundation Europe, 2019, S.8).

2.5.2 Veröffentlichung von OSS in der Schweiz

In der Schweiz geben Bund, Kantone und Gemeinden jährlich rund 3 Milliarden Franken für die öffentliche Informatik aus. Durch eine engere Zusammenarbeit zwischen den Behörden bei der Entwicklung und Wartung, könnte ein Grossteil dieser Ausgaben vermieden werden. OSS bietet die Möglichkeit, nicht nur bestehende Software kostengünstig zu nutzen, sondern auch bestimmte Fachanwendungen gemeinsam mit anderen öffentlichen Stellen zu entwickeln (Jost, Battagliero, Kohli, Sancar, & Sollberger, 2013, S.2).

Ein Beispiel für die Förderung der Zusammenarbeit ist das OSS-Projekt *«OpenJustitia»* vom Bundesgericht. Die selbst entwickelte, auf Java-basierte Geschäftsverwaltungslösung wurde 2011 unter einer OSS-Lizenz veröffentlicht (inside-it.ch, 2011). Das Ziel des Bundesgerichts war, die Zusammenarbeit mit anderen nationalen und kantonalen Gerichten zu ermöglichen und so die

Entwicklungskosten zu sparen. Dieses Vorhaben wurde jedoch nicht von jeder Seite toleriert. So lehnte beispielsweise das Softwareunternehmen Weblaw die Freigabe ab, da dessen Produkt durch Open Justitia direkt konkurriert wird. Sie argumentierten, dass das Bundesgericht den Softwaremarkt verzerrt, indem es das Geld der Steuerzahler verwendet (Free Software Foundation Europe, 2019, S.8). Obwohl Weblaw gegen die Veröffentlichung von Open Justitia war, trat sie als aktives Mitglied der Open Source Community von OpenJustita bei, welche aus insgesamt 17 Mitgliedern besteht (Open Justitia, 2019).

Weil Unklarheit herrschte, fanden auf nationaler und kantonaler Ebene verschiedene Vorstösse statt. Auf nationaler Ebene wollten gewisse Parlamentarier mit einer Interpellation im Jahre 2012 die Freigabe von OSS durch Bundesbehörden explizit erlauben (Weibel, 2012). In der Folge lies der Bundesrat ein Gutachten durch die Verwaltungswissenschaftler Georg Müller und Stefan Vogel erstellen. In einer 36-seitigen Publikation vertreten die Autoren die Meinung, dass Behörden untereinander ohne Weiteres Software-Entwicklungen austauschen dürfen. Für die öffentliche Freigabe als OSS würde allerdings eine gesetzliche Grundlage benötigt, weil jegliche Erbringung marktfähiger Leistungen an Dritte als wirtschaftliche Tätigkeit zu qualifizieren sei. Die kostenlose staatliche Softwareabgabe an Dritte ohne gesetzliche Grundlage müsse als Wettbewerbsverzerrung angeschaut werden (Free Software Foundation Europe, 2019, S. 18; Müller & Vogel, 2014, S.33ff.).

Als Antwort auf das Gutachten wurde Ende 2014 das Postulat 14'4275 *"Wie kann die Freigabe von Open-Source-Software durch die Bundesverwaltung explizit erlaubt werden?"* vom Nationalrat Balthasar Glätti eingereicht und im Jahr 2015 vom Nationalrat angenommen. In diesem Postulat wird der Bundesrat erstens beauftragt, zu prüfen, ob bestehende Finanzhaushaltgesetzte ergänzt werden müssten, so dass die Freigabe von Quellcodes durch den Bund explizit erlaubt wäre und gegebenenfalls in einem Zweiten Schritt die entsprechenden Anpassungen vorzuschlagen, um die OSS-Strategie der Bundesverwaltung umsetzen zu können (Glätti, 2014; Urech, 2015).

Parallel kämpften Parlamentarier in Bern auch auf kantonaler Ebene für die Freigabe von OSS durch den Kanton. So wurde eine Motion 2013.0783 *«Synergien beim Software-Einsatz im Kanton Bern nutzen»* eingereicht (Jost et al., 2013). Im Anschluss wurde ein zweites Rechtgutachten vom Kanton Bern in Auftrag gegeben. Das Gutachten von Prof. Dr. iur. Tomas Poledna, Prof. Dr. Simon Schlauri und MLaw

Samuel Schweizer kam zu einem anderen Entschluss (Poledna et al., 2017). In ihren Schlussfolgerungen kommen sie zum Ergebnis, dass es nicht notwendig sei, ein separates Gesetz zu erlassen, das es Regierungsbehörden erlaubt, Freie Software zu veröffentlichen (Poledna et al., 2017, S.173ff.). Die Autoren sind der Ansicht, dass in den meisten Fällen die unentgeltliche Veröffentlichung reiner Quellcodes keine marktfähige Leistung sei, die einer spezifischen Regulierung bedarf, weil es erst komplementäre Dienstleistungen wie Integration, Wartung oder Support braucht, um ihn nutzen zu können (Poledna et al., 2017, S.28). Daher kann die Veröffentlichung von OSS durch die öffentliche Hand nicht als schwerer Eingriff in die Wirtschaftsfreiheit eingestuft werden, welche ein formelles Gesetz verlangen würde, weil privaten Konkurrenten das Wirtschaften nicht verunmöglicht wird (Poledna et al., 2017, S.174). Hinzu kommt, dass durch die Veröffentlichung von OSS durch Verwaltungsstellen der freie Wettbewerb zwischen Informatikunternehmen gefördert wird, weil sie neue Möglichkeiten und Nachfragen nach kommerziellen Dienstleistungen wie Beratung, Einführung, Wartung, Schulung, Weiterentwicklung usw. um die Open-Source-Produkte schafft (Poledna et al., 2017, S.25). Anfang 2018 wurde die bestehende Verordnung über die Informations- und Telekommunikationstechnik der Kantonsverwaltung (ICTV) des Kanton Berns offiziell erweitert, indem ausdrücklich darauf hingewiesen wurde, dass die Freigabe eines eigenen Quellcodes unter einer OSS-Lizenz zulässig ist (Regierungsrat des Kantons Bern, 2018).

Auf Bundesebene fehlt eine solche Regelung. 2017 wurde ein Bericht als Antwort auf das oben genannte Postulat 14'4275 veröffentlicht. Der Bericht zeigt auf, dass der Bedarf in der Bundesverwaltung, OSS freizugeben, als gering eingeschätzt wird. Ob die kostenlose Freigabe von OSS durch den Bund ohne Gesetzesänderung zulässig ist, wurde aufgrund der zwei erwähnten unterschiedlichen Rechtsgutachten nicht abschliessend beantwortet. Damit eine einheitliche Rechtsanwendung in der Bundesverwaltung gewährleistet wird, hat der Bundesrat das Eidgenössisches Finanzdepartement und das Eidgenössischen Justiz- und Polizeidepartement beauftragt, die offenen Fragen gemeinsam abzuklären und basierend darauf die allenfalls notwendigen gesetzlichen Grundlagen zu erarbeiten (Schweizerische Eidgenossenschaft, 2017).

Ein Kernteam bestehend aus Vertreten des Informatiksteuerungsorgans des Bundes (ISB), des Bundesamtes für Bauten und Logistik (BBL) und des Bundesamtes für Informatik und Telekommunikation (BIT) wurde für eine Situationsanalyse bzgl. OSS in der Bundesverwaltung gebildet. Begleitet wurde der Prozess von der Forschungsstelle Digitale Nachhaltigkeit der Universität Bern. Die Situationsanalyse mittels qualitativen Interviews und Workshops mit Vertretern von Leistungsbezügern und Leistungserbringer hat ergeben, dass OSS in der Bundesverwaltung breit eingesetzt wird, jedoch noch Unsicherheiten bestehen. Deshalb wurde ein strategischer Leitfaden *«Open Source Software in der Bundesverwaltung»* ausgearbeitet. Darin sind fünf Ziele enthalten, welche mit sieben Massnahmen und konkreten Handlungsempfehlungen erreicht werden sollen. Ein Punkt im Leitfaden betrifft die Freigabe von OSS durch die Bundesverwaltung, welche für eine effiziente Nutzung von OSS in der Softwareentwicklung erforderlich ist. Für die Freigabe fehlt jedoch eine klare Rechtsgrundlage, weshalb nur gewisse Bundesämter wie swisstopo, Bundesgericht oder die Arbeitslosenversicherung bereits Projekte auf GitHub veröffentlicht haben und laufend weiterentwickeln. Ein Entscheid des Bundesrates ist aktuell noch ausstehend. Im Anschluss an diesen Entscheid ist einer der ersten Schritte, die Richtlinien für das Beitragen zu bestehenden Projekten und Veröffentlichen von neuen Open Source Projekten durch die Bundesverwaltung zu verfassen (Informatiksteuerungsorgan des Bundes, 2019).

2.6 Zusammenarbeit mit Externen

Zu einem früheren Zeitpunkt war es nicht notwendig, anzugeben, dass ein Open-Source-Projekt von der Community verwaltet wird, weil ein von der Community verwaltetes Entwicklungsmodell als Teil der Definition eines Open-Source-Projekts betrachtet wurde (O'Mahony, 2007, S.140). Die von der Community verwalteten OSS-Projekte, weisen fünf Kernprinzipien auf, welche die *Governance-Struktur* bestimmen: Unabhängigkeit, Pluralismus, Repräsentation, dezentrale Entscheidungsfindung und autonome Beteiligung (O'Mahony, 2007, S.145). Ein von Unternehmen getriebenes OSS-Projekt kann durch das Gegenteil dieser Prinzipien definiert werden: Abhängigkeit von einem einzelnen Sponsor, Dominanz eines Unternehmens, unbestrittene Kontrolle durch einen Sponsor, zentralisierte Entscheidungsfindung durch die Unternehmensleitung und streng begrenzte

Beteiligung. Die zwei beschriebene Arten von Projekten sind als Extreme zu verstehen und in der Praxis dominieren viele intermediäre Formen (Stuermer, 2009, S4.f.).

Der Begriff OSS steht nicht im Widerspruch zu kommerzieller Software. OSS kann auch kommerziell entwickelt, vertrieben und eingesetzt werden. Der Unterschied zu *Closed Software* ist, dass OSS auch nicht-kommerziell entwickelt wird. Darüber hinaus kann es vorkommen, dass Unternehmen und Privatentwickler zeitgleich am selben Produkt arbeiten (Greve, 2016, S.31).

2.6.1 Einfluss von IT Unternehmen in OSS-Projekten

Auf der Grundlage von verfügbarem empirischen Material, Fallstudien und Übersichtsdarstellungen hat Schrape (2016) den Einfluss von Unternehmen und deren Koordinationsformen mit der Community in vier idealtypische Varianten von Open-Source-Projekten unterteilt, welche in Abbildung 4 illustriert und im Folgenden erläutert werden.

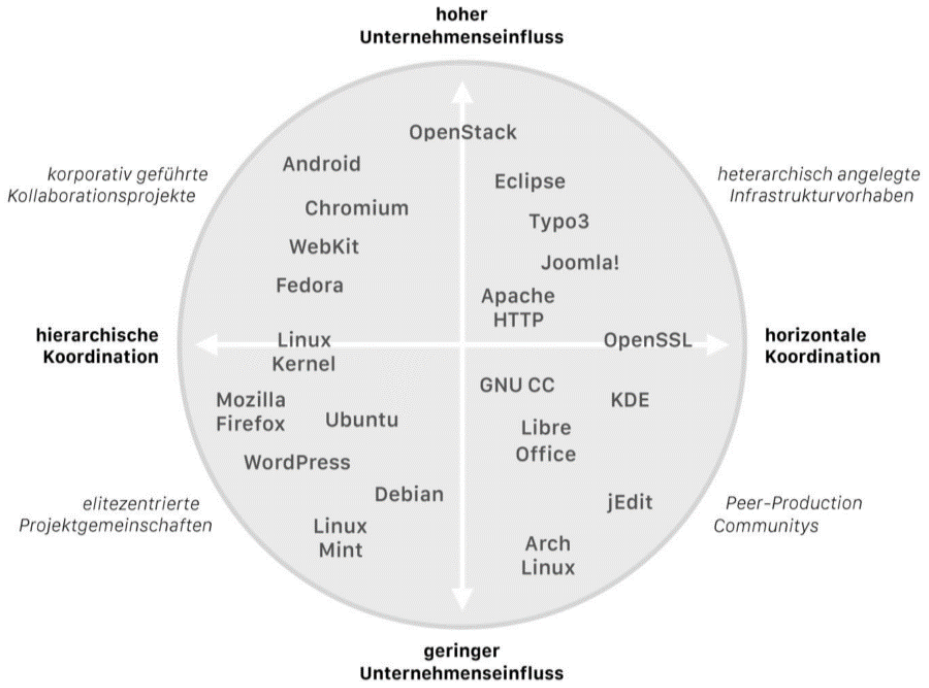

Abbildung 4: Idealtypische Varianten von Open-Source-Projekten

Korporativ geführte Kollaborationsprojekte haben das Ziel ein gemeinsames Produkt und gemeinsame Plattform zu erarbeiten. Korporativ geführte Kollaborationsprojekte zeichnen sich durch deutliche Hierarchien auf Arbeitsebene aus und sind durch ein anleitendes Unternehmen gesteuert. Die Communiies bestehen vor allem aus angestellten Entwicklern, welche auf individueller Ebene oder als

explizite Unternehmensvertreter problemzentriert zusammenarbeiten. Berühmte Beispiele solcher Kollaborationsprojekte sind *Android* und *WebKit*, bei welchen die strategische Kontrolle eindeutig bei Google bzw. Apple liegt (Schrape, 2016, S.50ff.).

Heterarchisch angelegte Infrastrukturvorhaben verspüren auch einen hohen Unternehmenseinfluss, unterliegen jedoch nicht deren direkter Kontrolle, sondern werden in der Regel von einer Stiftung getragen. In solchen Infrastrukturvorhaben liegt der Fokus auf Entwicklung von softwaretechnischen Standards und Infrastrukturen. Berühmte Beispiele sind die integrierte Entwicklungsumgebung Eclipse oder die Webserver-Software wie der *Apache HTTP Server*. Weitere Merkmale solcher Projekte sind deren längere Existenz, eine stabile Code-Basis und ein dadurch geringeres Aktivitätsniveau als es bei neueren Projekten anzutreffen ist (Schrape, 2016, S.56f.).

Als drittes gibt es noch die elitezentrierte Projektgemeinschaften, welche einen moderateren Einfluss von Unternehmen haben. Geführt werden solche Projekte vom Gründer, einem Führungsteam oder einem gewählten Projektleiter. Berühmte Beispiele solcher Projekte sind der *Linux Kernel* oder *Mozilla* (Schrape, 2016, S.60ff.)

Peer-Production-Communities sind marktunabhängig und weisen eine gleichberechtigte Zusammenarbeit unter intrinsisch motivierten Entwicklern aus. Ab einer gewissen Marktrelevanz sind jedoch abgestufte Führungsstrukturen und ein Pool an korporativen Stakeholdern zu beobachten. Berühmte Beispiele eines solchen Projekt ist die Bürosoftware *LibreOffice* (Schrape, 2016, S.66ff.).)

Die Gemeinsamkeit aller vier idealtypischen Varianten von OSS-Projekten ist, dass die quelloffenen Lizenzmodellen die jeweiligen Projekten vor der Proprietarisierung schützen (Schrape, 2016, S.75).

2.6.2 Open Source Community Strategie

Nach Dahlander und Magnusson (2008) ist die Nutzung von Communities eine Möglichkeit für Unternehmen, die Ressourcen für Innovationen zu erhöhen indem sie auf kreativen Ideen von Personen außerhalb des Unternehmens zurückgreifen. Dies geschieht jedoch nicht spontan, sondern neue Strategien und Arbeitsweisen sind erforderlich, um eine gute Übereinstimmung zwischen dem, was das Unternehmen will und was für Ressourcen und Fähigkeiten in seinem externen Umfeld verfügbar sind, zu schaffen.

Als erstes müssen Unternehmen entscheiden, ob sie auf bestehende Communities zugreifen oder ihre eigenen Communities gründen. Zudem müssen sie entscheiden, ob sie einen adaptiven Ansatz wählen, bei welchem der Fokus auf der Nutzung des in der Community entwickelten Materials liegt und das Unternehmen versucht dieses Material mit internen Komponenten zu integrieren. Oder sie übernehmen eine aktive Rolle in der Community und versuchen die Entwicklung der Community zu beeinflussen. Jedoch sind die Mittel, um Einfluss ausserhalb des Unternehmens zu erlangen unterschiedlich gegenüber interner Einflussnahme. Daher ist es notwendig, dass Unternehmen eine ganze Reihe subtiler Kontrolltechniken anwenden, um die Mitglieder der Community zu motivieren. Diese Kontrolltechniken sollten nicht hierarchisch sein, sondern finanzielle Anreize oder intellektuelle Herausforderungen schaffen. Vor allem letzteres stellt das Unternehmen vor eine Herausforderung, weil die Entwicklung von Software auch die Durchführung von weniger spannenden, routinemässigen Aufgaben beinhaltet. Das Unternehmen kann sich nicht auf hierarchische Kontrollen oder vertragliche Vereinbarungen verlassen, weshalb es eine Möglichkeit ist, dass das Unternehmen eine Arbeitsteilung vornimmt, indem die Community die kreative Arbeit leistet und das Unternehmen die routinemässigeren Aufgaben übernimmt. Unternehmen müssen strategische Entscheidungen darüber treffen, was intern unternommen wird und wann sie sich auf externe Quellen verlassen können (Dahlander & Magnusson, 2008, S644ff.).

Neben dem Identifizieren von nützlichem externem Wissen, müssen Unternehmen auch in der Lage sein, externes Wissen zu assimilieren und anzuwenden, weil sich die Rolle der Unternehmen von der internen Entwicklung und Fertigung zur Zusammenführung von Wissen und Komponenten verlagert. Die Grundlage für die Entwicklung und Aufrechthaltung von Wettbewerbsvorteilen ist, dass Unternehmen ihre Umwelt effizient und effektiv analysieren, Entwicklungen ausserhalb ihres Kernbereichs bewerten und eine schnelle und nahtlose Integration des externen Wissens ermöglichen (Dahlander & Magnusson, 2008. S.645f.; Porter, 1996, S.1).

Was die Koordination und Kontrolle von Communities anbelangt, müssen Unternehmen andere Wege als die traditionellen Kontrollmittel finden, um Beiträge zu ihren OSS-Projekten zu erhalten. Unternehmen sollten einen symbiotischen Ansatz verfolgen, indem sie ihr Wissen zur Verfügung stellen und die Normen und Werte von

der Community respektieren, damit sowohl die Gemeinschaft als auch das Unternehmen von der Beziehung profitieren (Dahlander & Magnusson, 2005).

Wenn Unternehmen eine eigene Community aufbauen und mit ihr von Anfang an entwickeln wollen, dann muss das Unternehmen ein plausibles Versprechen abgeben können. Die Software muss nicht besonders gut funktionieren, sondern es kann grob, fehlerhaft, unvollständig und schlecht dokumentiert sein. Es muss jedoch lauffähig sein, um potenzielle Mitentwickler davon überzeugen zu können, dass es in absehbarer Zeit zu etwas Ordentlichem entwickelt werden kann. Linux ist ein gutes Beispiel, welches mit einem attraktiven Basisdesign an die Öffentlichkeit ging (Raymond, 1999, S.37).

Die Geschichte von OSS zeigt auf, dass sich OSS in den letzten 50 Jahren rasant entwickelt hat und geschäftstauglich wurde. Auf der anderen Seite fand auch eine Annäherung von Unternehmen an OSS und OSS-Communities statt. Frühere Studien haben vor allem IT-Unternehmen analysiert. In den folgenden Kapiteln liegt der Fokus auf nicht IT-Unternehmen und Behörden in der Schweiz.

3 Forschungsdesign

Weil die Veröffentlichung von OSS von nicht IT-Unternehmen und Behörden eine neue Erscheinung ist, gibt es kaum Literatur oder Forschung zu diesem Thema. Deshalb wurde für die vorliegende Arbeit ein explorative Fallstudienansatz gewählt, um zu untersuchen, warum nicht IT-Unternehmen und Behörden in der Schweiz einzelne Codes oder eine ganze Software unter einer OSS Lizenz veröffentlichen oder planen zu veröffentlich, welche strategische Bedeutung OSS für sie hat, wie sie dabei vorgehen und wie sie mit Externe zusammenarbeiten. Wie Yin (2003) beschreibt, eignet sich der Fallstudienansatz, wenn der Fokus auf einem zeitgenössischen Phänomen in einem realen Umfeld liegt und *«warum»* oder *«wie»* Fragen beantwortet werden sollen.

3.1 Auswahl der Fallstudien

Die zu untersuchenden Fälle wurden nicht zufällig ausgewählt, sondern die Auswahl erfolgte auf der Grundlage der Erwartungen an ihren Informationsgehalt. Dafür wurde der Ansatz der maximalen Variation der Fälle gewählt, um Informationen über die Bedeutung verschiedener Umstände für das Ergebnis zu erhalten (Flyvbjerg, 2006, S.34). Ein Auswahlkriterium war, dass die zu untersuchenden Fälle ihre Kernkompetenz nicht in der Softwareentwicklung haben und nicht aus der IT-Industrie stammen. Ein weiteres Auswahlkriterium war, dass die zu untersuchenden Fälle einzelne Codes oder ein Projekt unter einer OS-Lizenz veröffentlicht haben oder eine Veröffentlichung planen.

So wurden vier Unternehmen aus drei verschiedenen Branchen (SRF Data aus der Medienbranche, SIX Group aus der Finanzbranche und Baloise und Helvetia Versicherung aus der Versicherungsbranche) und zwei Behörden (das Kompetenzzentrum für Informations- und Kommunikationstechnik der Verwaltung des Kantons Bern und das Bundesamt für Landestopografie) ausgewählt. Die Unternehmen in der Versicherungsbranche unterscheiden sich bezüglich der Maturität inwieweit OSS in ihrem Unternehmen schon adaptiert wurde. Während die Baloise eigene Projekte bereits veröffentlicht hat, ist die Helvetia in der Planung für Veröffentlichungen.

3.2 Datensammlung und Analyse

Die sechs Fallstudien basieren auf semi-strukturierte Interviews (Appendix1) Vor der Befragung erhielten die Interviewpartner einen Leitfaden mit den wichtigsten Fragen. Danach wurden mit acht verschiedenen Stakeholdern, welche sich in ihrem Umfeld aktiv für die Förderung für OSS-Projekten befassen, sechs offen Interviews durchgeführt. Das Ziel der Interviews war, bei bestimmten Themen detailliertere Nachfragen stellen zu können und auf diese Weise den Sachverhalt tiefgründiger zu erfassen. Tabelle 2 zeigt die wichtigsten Variablen der untersuchten Fälle, ihre Vertreter soie die Interviews selbst.

Markus Tiede von der Baloise Groupe, Timo Grossenbacher von SRF, Michael Stolz und Daniel Zuck von der Six-Group, Henning Henkel und Tobias Denzler von der Helvetia Versicherung, Thomas Joos vom Amt für Informatik und Organisation des Kantons Bern und David Ösch vom Bundesamt für Landestopografie swisstopo.

Fälle	SRF Data	Baloise- Groupe	Helvetia Ver-sicherung	SIX -Group	KAIO	swisstopo
Branche / Kategorie	Medien / Journalism us	Versicherung	Versicherung	Finanzen	Kantonale Behörde	Bundes Behörde
Intervie-partner (Personen-nummer)	Timo Grossen-bacher (F1)	Markus Tiede (F2)	Henning Henkel (F3a); Tobias Denzler (F3a)	Michael Stolz (F4a); Daniel Zuck (F4b)	Thomas Joos (F5)	David Ösch (F6)
Rolle	Daten-journalist	Software-entwickler (Integration Services Team)	Automation Solution Architect; Plattform Solution Architect	Software-engineer im Testing Techlead für die Linux Plattform.	Servicemanager , Fach-bereich Software	Projekt-leiter für geo.admin.ch
Datum	2019-01-17	2019-01-16	2019-01-16	2019-01-17	2019-01-25	2019-01-24
Dauer	63 min	55 min	75 min	60 min	54 min	55 min
Sprache	Schweizer-deutsch	Deutsch	Deutsch und Schweizer-deutsch	Deutsch und Schweizer-deutsch	Schweizer-deutsch	Schweizer-deutsch
Ort	Zürich, CH	Basel, CH	Basel, CH	Zürich, CH	Bern, CH	Bern, CH

Tabelle 2: Informationen zu den untersuchten Unternehmen und Personen

Alle Interviews wurden mit Hilfe von MAXQDA, einer Software zur computergestützten qualitativen Daten- und Textanalyse, transkribiert und analysiert. Die Transkription erfolgte nach einer sauberen wortgetreuen Abschrift. Alle Verzögerungslaute wie zum Beispiel *«äh»* wurden weggelassen und Dialekte wurden ins Schriftdeutsche übersetzt. Das Resultat der Transkription ist ein zusammenhängender Text, der einfach zu verstehen ist, aber den ursprünglichen Wortlaut und die grammatikalische Struktur wiedergibt (Mayring, 2014, S.45). Um die Validität der Ergebnisse zu erhöhen, erhielten die Befragten die Transkription der Interviews und wurden gebeten, gegebenenfalls Verbesserungen vorzuschlagen.

Die Auswertung der transkribierten Interviews erfolgte anhand der qualitativen Inhaltsanalyse. Dabei wurde das Verfahren der induktive Kategorienbildung angewendet. Es wurden nur die Textausschnitte der Interviews berücksichtigt, die für die übergeordneten Forschungsfragen dieses wissenschaftlichen Beitrags relevant sind. Diese Technik zeichnet sich durch die Bildung von einzelnen Kategorien aus, die direkt aus dem Material stammen und für thematische Verallgemeinerungen genutzt werden. Dies diente als Orientierung für die Analyse und Auswertung des aufbereiteten Materials (Mayring, 2014, S.S.79ff.).

4 Analyse der Fallstudien

In den folgenden Abschnitten wird in einem ersten Schritt jeder einzelne Fall als eigenständige Einheit analysiert, um einzigartige Muster jedes Falles erkennen zu können. In einem zweiten Schritt werden die wichtigsten Ergebnisse anhand einer fallübergreifenden Analyse zusammengefasst (Eisenhardt, 1989, S.540). Jeder Fall wird durch eine kurze Beschreibung des Unternehmens oder der Behörde eingeleitet. Im Anschluss werden für jeden Fall die Treiber der Idee, der Nutzen und die Risiken der Veröffentlichung, die strategische Bedeutung von OSS, die interne Organisation, die Prozesse, die Richtlinien, sowie die Zusammenarbeit mit Partnern einzeln analysiert. Die Zitate der Befragten werden mit der Personennummer identifiziert (siehe Tabelle 2).

4.1 SRF Data Fallstudie

Das Unternehmen Schweizer Radio und Fernsehen (SRF) ist ein unabhängiges Medienhaus, welches unter anderem die freie Meinungsbildung durch sachgerechte Information fördert. Das SRF ist Teil von der SRG, dem grössten Unternehmen für elektronische Medien in der Schweiz, welches politisch und wirtschaftlich unabhängig ist. Als Non-Profit-Unternehmen finanziert sich die SRG zu ungefähr 75 Prozent über Gebühren, zu rund 25 Prozent über kommerzielle Tätigkeiten. Das SRF beschäftigt insgesamt 2100 Mitarbeiterinnen und Mitarbeiter (Srf.ch, 2019).

Seit 2014 gibt es bei der SRF das SRF Data Team, mit drei Kernmitarbeiter, welche Datenjournalismus betreiben. Die Aufgabe des Teams besteht darin Daten mithilfe von Automatisierung und Programmierung journalistisch auszuwerten. Dazu gehört auch die Visualisierung von Daten (F1). Neben den Artikeln welches das SRF Data Team produziert, werden seit 2016 auch die Datensätze und Codes auf *GitHub* veröffentlicht (SRF Data, 2019b).

4.1.1 Treiber für die Veröffentlichung von Open Source Software

Die Inspiration für die Veröffentlichung von eigenen Projekten kam aus dem angelsächsischen Raum. Vor allem eine bestimmte amerikanische Plattform, *FiveThirtyEight*, wurde im Gespräch als Inspirationsquelle genannt. Diese Plattform betreibt Datenjournalismus und hat damit begonnen, einen Teil der Daten, die sie selbst aufbereitet haben, auf *GitHub* in einem einzigen *Repository* zu veröffentlichen.

Die Plattform beschränkt sich jedoch auf die Veröffentlichung von Daten, ohne der Codes (F1).

Die Idee, eigene Projekte zu veröffentlichen, kam vom SRF Data Team, mit dem Ansporn, es besser machen zu wollen als im angelsächsischen Raum. Dabei sollten, im Gegensatz zur Vorgehensweise der oben erwähnten amerikanischen Plattform, nicht nur Daten, sondern auch Codes veröffentlicht werden, um eine bessere Reproduzierbarkeit und Dokumentation zu erreichen. Einen weiteren Auslöser für das Veröffentlichen von eigenen Projekten sah das SRF Data Team im Öffentlichkeitsauftrag des SRF. Es passe zur Logik des Service public, dass das Veröffentlichen von Codes auch der Allgemeinheit zu Gute komme (F1).

4.1.2 Nutzen und Risiken eigene Projekte zu Veröffentlichen

Als Hauptgrund, wieso eigene Daten und Codes veröffentlicht werden, nennt SRF Data die Schaffung von Transparenz. Es wird als sehr wichtig betrachtet, dass etwas nachvollziehbar und transparent sei. Gerade beim Schreiben von Codes und beim Analysieren von Daten treffe man zahlreiche Annahmen und Entscheidungen. Diese Entscheidungen sollen transparent gemacht werden, damit es möglich wird, diese zu hinterfragen und zu kritisieren (F1).

Ein weiterer genannter Nutzen ist die Reputation. Dabei soll deutlich werden, dass die Arbeit, die geleistet wird, auch anderen zugutekommt. Die Aktivitäten der SRF Data sei in der Open Data Community und unter Journalisten bekannt und würde von diesen geschätzt. Dies dient unter anderem als gutes Aushängeschild (F1).

Auf der anderen Seite wurde das Risiko genannt, dass das Veröffentlichen (zu) viel Zeit in Anspruch nehmen könnte und sich deshalb amortisieren sollte. Dazu wird ein Beispiel genannt, wo eine Veröffentlichung, sprich die Entwicklung der betreffenden Templates, ungefähr 100 Stunden in Anspruch nimmt. Je mehr Projekte dann mit diesen Templates veröffentlicht werden können, desto eher amortisierten sich die dabei entstandenen Kosten (F1).

Das Risiko, durch die Veröffentlichungen einen Wettbewerbsnachteil zu erleiden, wird als gering eingestuft. Dies vor allem aus dem Grund, da die Daten zu einem Zeitpunkt veröffentlicht werden, wo daraus schon eine Story gemacht wurde (F1).

4.1.3 Strategische Bedeutung

Die strategische Bedeutung von OSS für SRF als Unternehmen ist relativ gering und wird vom Management nicht aktiv gefördert. Vielmehr wird etwas in der Nische des Datenjournalismus publiziert. SRF als Gesamtorganisation besitzt keine übergreifende, einheitliche OSS-Strategie. Die interne Praxis des Datenjournalismus ist dem Management aber bekannt. Es hat zu dieser Praxis nie Widerspruch innerhalb der Organisation gegeben (F1).

Die Strategie vom SRF Data Team besteht darin, dass wo möglich nur OSS eingesetzt wird und proprietäre Software fast nie verwendet wird (F1).

Es werden nur Daten und Codes veröffentlicht, welche für Rechercheergebnisse relevant sind. *Frontend Codes* zum Beispiel werden nicht veröffentlicht, da es dabei nicht um die Auswertung und Analyse von Daten geht. *Fronted Codes* betreffen somit nicht direkt die Gewinnung von Rechercheergebnissen. In solchen Fällen wird Transparenz deshalb als weniger wichtig eingeschätzt, wie bei den Auswertungen (F1).

4.1.4 Interne Organisation

Die hierarchische Organisationsstruktur innerhalb des Teams ist flach. Es gibt keinen offiziellen Chef und keine festen Rollen im Team. Jede Person kann verschiedene Aufgaben übernehmen, wie Artikel oder Codes schreiben, oder recherchieren. Die Zusammenarbeit erfolgt über *GitHub*, wobei jeder Mitarbeiter mit einem privaten Account Mitglied ist. SRF Data stellt dabei keinen Account im normalen Sinne dar, sondern eine Organisation, wo andere Accounts dazukommen. Neben den drei Kernmitarbeitern haben andere temporären Zugriff auf den GitHub Account. Dazu gehören z.B. Praktikanten oder Redaktionsgäste. Zum Teil wird auch mit externen Firmen zusammengearbeitet, vor allem im Bereich der Datenvisualisierung (F1).

Die Mitarbeitenden werden intern informell geschult. Jeder der Mitarbeitenden arbeitet mit *GitHub* und kennt sich demnach damit aus. Wenn Redaktionsgäste oder Praktikanten kommen, werden diese am Anfang in diesem System geschult. Die Mitarbeitenden befassen sich auch ausserhalb der Arbeitszeit mit OSS und machen teilweise auch private eigene Projekte. Das Unternehmen profitiert vom privaten Engagement, indem durch das Projekt die ganze Veröffentlichung auf *GitHub* automatisiert wurde. Für die Entwicklung während der Arbeitszeit brauchte es keine

explizite Genehmigung. Ein Teil der Entwicklungsarbeit wird in der Freizeit, ein Teil während der Arbeitszeit erledigt, wenn es für die SRF einen Nutzen bringt (F1).

4.1.5 Richtlinien und Prozesse

Wie externe Komponenten eingesetzt werden, wird durch informelle Richtlinien vorgegeben. Als Beispiel wurde dafür genannt, dass bei R *Packages* genommen werden, von denen man weiss, dass sie qualitativ gut und getestet sind. Auch für die Veröffentlichung von Daten und Codes gibt es teaminterne informelle Richtlinien. Zum Beispiel steht in diesen Richtlinien geschrieben, dass ein Code kommentiert werden sollte und dass keine sensitiven Informationen, wie z.B. Passwörter, im Code stehen dürfen (F1).

Alle Projekte auf *GitHub* sind unter *«Creative Commons Namensgebung, Weitergabe unter gleichen Bedingungen»* lizenziert. Der Grund, wieso diese Lizenz ausgewählt wurde, war die «Einfachheit». Dabei wurde auch die Rechtsabteilung involviert, welche der Meinung war, dass diese Lizenz Sinn ergeben würde. Zudem war sie der Meinung, dass ein Haftungsausschluss beifügt werden sollte (F1).

Das Ziel des SRF Data Teams ist es, die Daten und Codes immer zu veröffentlichen. Ausser gewichtige Gründe sprechen gegen eine Veröffentlichung. Der Hauptgrund, etwas nicht zu veröffentlichen, besteht im Einhalten des Datenschutzes. Ausserdem gibt es Situationen, wo Daten exklusiv zugespielt werden. Unter diesen Umständen kommt eine Veröffentlichung nicht in Frage. Ein weiterer Grund nicht zu veröffentlichen ist, wenn die Auswertung trivial ist und niemand Interesse an der Veröffentlichung des betreffenden Codes oder Daten hat. Die Datenaufbereitung und das Schreiben der Codes findet nicht öffentlich statt. Da sich SRF Data im investigativen Bereich bewegt, ist es in ihrem Interesse, dass nicht öffentlich wird, woran sie gerade arbeiten. Aus diesem Grund werden auch keine unfertigen Sachen veröffentlicht (F1).

Die Qualität der Datenaufbereitung steht im Fokus. Dabei ist es von hoher Bedeutung, dass die veröffentlichten Informationen nachvollziehbar und gut dokumentiert sind, damit jemand mit Grundkenntnissen im Programmieren diese nutzen kann. Deshalb gibt es vor jeder Veröffentlichung einen *Review*. Dabei wird das Vier-Augen-Prinzip angewandt und es wird versucht, den betreffenden Code nochmals auf einer anderen Architektur laufen zu lassen. Es werden keine *Code*

Scanning Tools verwendet, weil hauptsächlich mit OSS-Komponenten gearbeitet wird, welche mit *MIT* -oder *Creative Commons* Lizenzen lizenziert sind (F1).

Die Daten und Codes werden mit der Publikation des Artikels veröffentlicht. Somit können Personen, welche einen Artikel auf *Srf.ch* gelesen, oder einen Bericht im *10 vor 10* gesehen haben, die damit zusammenhängenden Daten von Anfang an finden. Die Veröffentlichung auf *GitHub* wurde automatisiert, indem ein *Deploymentscript* das R *Markdown* in ein HTML Datei umwandelt und dieses automatisch auf GitHub hochlädt. Auf die Daten kann auf drei verschiedene Arten zugegriffen werden. Einerseits findet sich der Code in einem *Public GitHub Repository*, inklusiv der Daten. Zudem ist das *Markdow*n als *HTML File* auf der *GitHub Page* publiziert. Als dritte Möglichkeit wird zusätzlich ein gezippter Ordner geschaffen (F1).

Für externe Anfragen zu den veröffentlichten Daten und Codes gibt es keinen Prozess. Diese werden individuell beantwortet. Allerdings sind die Ressourcen dafür begrenzt und Grundkenntnisse werden vorausgesetzt. So ist es z.B. aus Ressourcengründen nicht möglich, Grundkenntnisse des R zu vermitteln (F1).

4.1.6 Zusammenarbeit mit anderen

Die Zusammenarbeit mit externen hält sich in Grenzen. Die einzigen Partner, mit denen zusammen auf *GitHub* entwickelt wird, sind Firmen, welche bei der Datenvisualisierung helfen. Dabei wird geschaut, wer die Anforderungen erfüllen kann. Im Anschluss folgt ein *Compliance* orientierter Beschaffungsprozess. Das SRF Data Team hat eine Twitter *Community*. Aus Ressourcengründen wird jedoch kein aktives *Community Management* betrieben. Die bestehenden Verbindungen werden als eher informell und ohne starke Berührungspunkte bezeichnet. Da das Team nur aus drei Personen besteht, würde ein aktiveres *Community Management* den aktuellen Rahmen der Möglichkeiten des SRF Data Teams sprengen (F1).

4.2 Baloise Group Fallstudie

Die Baloise Holding AG ist ein Versicherungskonzern und heute als Baloise Group bekannt. Die Baloise agiert als Anbieter von Präventions-, Vorsorge-, Assistance- und Versicherungslösungen. Die Kernmärkte der Baloise sind die Schweiz, Deutschland, Belgien und Luxemburg (Baloise.com, 2019a). In der Schweiz agiert die Baloise mit den Marken *«Basler Versicherung»* und *«Baloise Bank SoBa»*

Mit der dualen Kompetenz kann die Baloise in der Schweiz ihren Kunden Versicherungs- und Banklösungen aus einer Hand anbieten (Baloise.ch, 2019). Die Baloise setzt schon seit längerer Zeit OSS-Lösungen ein. Seit ein paar Jahren ist sie auf *GitHub* und besitzt einen Account als Organisation. Aktuell spielt sich bei der Baloise nach eigener Aussage alles auf *GitHub* ab. So gibt es auf *GibHub* Projekte und Organisationen. Die Organisation für die Basler Gruppe gibt es dabei schon relativ lang, seit ungefähr vier oder fünf Jahren (F2).

4.2.1 Treiber für die Veröffentlichung von Open Source Software

OSS im Namen der Baloise zu veröffentlichen wurde von den Mitarbeitenden getrieben. Die Idee kam mit der *«Agilisierung»* der Baloise vor fünf bis sechs Jahren, als zu den einzelnen Scrum-Teams, Expertengruppen etabliert wurden, die sich teamübergreifend austauschen können. In der Software Engineering Gruppe wurde ein *«Goldcard Konzept»* eingeführt, welches Innovationen fördern sollte, indem die Mitarbeitenden mehr Freiräume erhielten. Dies wurde zu einem gewissen Grad von Google kopiert. Die meisten Teams arbeiten in zwei Wochen Sprints. Innerhalb dieser 2 Wochen kann sich jeder Sprintteilnehmende einen halben Tag lang Zeit nehmen, bestimmte Dinge zu tun, so genannte *Goldcards*. In den Anfängen sind viele dieser *Goldcards* Ideen und Projekte auf *GitHub* gestellt worden (F2).

4.2.2 Nutzen eigene Projekte zu Veröffentlichen

Den Hauptnutzen in der Veröffentlichung von OSS sieht die Baloise in den Kostenersparnissen. Dies wird auch als Argument genannt, um die Akzeptanz dieser Vorgehensweise innerhalb des Unternehmens zu fördern (F2). Ein spezifisches Ziel ist dabei die Identifikation von anderen Unternehmen, die an der Teilnahme an OSS-Projekten interessiert sind. Dadurch erhofft sich die Baloise unter anderem Kostenersparnisse, weil Betriebs-, Wartungs- und Weiterentwicklungskosten aufgeteilt werden können (F2).

Einen weiteren Nutzen, welchen die Baloise in der Veröffentlichung von OS-Komponenten sieht, ist die Verbreitung und Bekanntmachung eigener Lösungen. Insbesondere beim API Management könnten OSS-Komponenten zur Diffusion eigener Lösungen beitragen (F2).

Die Arbeitgeberattraktivität ist ein weiterer Vorteil, welche die Veröffentlichung von Open Source Software mit sich bringt. Der Gesprächspartner

meinte, er selbst sei zur Basler gekommen, weil er gesehen habe, dass diese mit Open-Source arbeitet, was ihn persönlich motiviert habe (F2). Ziel der Baloise ist es, gemäss der neuen Strategie *«Simply Safe»,* bis 2021 zu den besten zehn Prozent der Arbeitgeber in der Branche zu gehören (Baloise.com, 2019b). Die Entwickler stehen im Kontakt mit der Personalabteilung, um aufzuzeigen, dass OSS ausschlaggebend sein kann bei der Rekrutierung neuer Mitarbeitender (F2).

Die Veröffentlichung von OSS bringt jedoch nicht nur bei der Rekrutierung einen Vorteil, sondern auch bei der Entwicklung bestehender Mitarbeitender. Schliesslich wird im Gespräch auch die Motivation geäussert, Neues zu lernen und sich weiterzuentwickeln und ein Stück des Gelernten zurückzugeben (F2).

Risiken durch die Veröffentlichung eigener Komponenten werden zurzeit als gering eingestuft. Dies aus dem Grund, dass bis anhin noch keine businesskritischen Komponenten veröffentlicht wurden. Dies könne sich jedoch ändern, sobald grössere strategische Projekte etabliert werden sollten (F2).

4.2.3 Strategische Bedeutung

Bei der Nutzung von OSS verfolgt die Baloise die Strategie, den kommerziellen Support von *Red Hat* - einem OSS-Dienstleister – in Anspruch zu nehmen. Eine offizielle OSS-Strategie für die Veröffentlichung gibt es bei der Baloise aktuell noch nicht. Sie befindet sich nach eigener Aussage in einem zu frühen Stadium der Entwicklung, so dass eine strategische Verankerung zum jetzigen Zeitpunkt keinen Sinn ergeben würde (F2).

Die Baloise hat bislang noch keine strategischen Projekte veröffentlicht. Bis anhin handelte es sich bei Veröffentlichungen eher um sogenannte Hilfsprojekte, die im Rahmen der normalen Softwareentwicklung entstanden sind (F2).

Ein erster Anlauf, ein strategisches OSS-Projekt gemeinsam mit anderen Unternehmen zu entwickeln, wurde letztes Jahr mit *Open Prevo* gestartet. *Open Prevo* wurde dabei als *Minimum Viable Product (MVP)* für sechs Wochen gestartet. Dies mit dem Ziel, ein *MVP* zu schaffen, mithilfe dessen gezeigt werden kann, dass eine gemeinsame Software zwischen verschiedenen Versicherungshäusern quelloffen geschrieben werden kann. Dabei ging es um den Transfer von Freizügigkeitsleistungen zwischen Pensionskassen in der Schweiz. Das Projekt wurde aktuell noch nicht veröffentlicht, es befindet sich im *Proposal* Status. Der wirtschaftliche Nachweis für das Projekt stellt aktuell eine Herausforderung dar für die Baloise. Als Grund dafür

wird genannt, dass diese Art der Vorgehensweise noch nicht definitiv Fuss gefasst habe in der Business Welt: *«Also zwischen technischem Wissen von den Engineers (…) und daraus wirklich eine strategische Investition zu bekommen. Dieser Schritt braucht mehr, als ein paar engagierte Engineers*» (F2).

Das Verständnis beim Management dafür, dass viele neue Projekte quelloffen gestartet werden sollen, ist vorhanden. Als grössere Herausforderung wird es empfunden, die Mitarbeitenden davon überzeugen zu können und diese in die Entwicklung mitzunehmen und aufzuzeigen, dass man von Methoden und Wissen ausserhalb des Basler Universums profitieren und lernen kann (F2).

4.2.4 Interne Organisation

Die Baloise hat kein offizielles OSS-Team nach aussen. Zwei Mitarbeiter treiben das Thema OSS jedoch, sowohl intern als auch extern und fungieren als Ansprechpartner. Dabei hat einer dieser zwei Mitarbeiter den Lead, der andere kümmert sich um Angelegenheiten im *Community* Bereich. Dazu gehört die Öffentlichkeitsarbeit, wie z.B. die Vertretung an Events. Dauerhaft arbeitet bei der Baloise noch niemand in OSS-Projekten. Die Zusammenarbeit in OSS-Projekten erfolgt über *GitHub*. Insgesamt sind es ungefähr 62 Personen, die bei der Baloise Teil der *GitHub* Organisation sind. Davon sind fünf bis zehn Mitarbeitende mittelmässig aktiv. Mittelmässig aktiv aus dem Grund, dass bislang ausschliesslich Projekte veröffentlicht wurden, die aus dem Kontext der oben genannten *Goldcards* Arbeitsprojekte entstanden sind (F2).

Das *«Goldcard Konzept»* führt dazu, dass sich die Mitarbeitenden zusammenschliessen und so unter anderem neue OS-Projekte entstehen. Durch dieses Summieren von *Goldcards* entsteht dann die Möglichkeit «*kleine bis mittelgrosse Projekte zu realisieren».* Die Entwickler treffen die Entscheidung hauptsächlich selbst, ob sie kleine Projekte veröffentlichen wollen. Für die Veröffentlichung dieser Projekte ist zurzeit keine Genehmigung erforderlich (F2).

Offizielle OSS-Schulungen gibt es bei der Baloise nicht. Die Holschuld ist eher aufseiten der Mitarbeitenden. Wenn sie Interesse haben, können sie sich jederzeit melden, dann wird individuell geschaut. Zudem gibt es bei der Baloise alle sechs Wochen interne Fortbildungstage, sogenannte *«Open X Days»*. Diese Tage bieten die Möglichkeit, wichtige Themen zu besprechen und weiterzutreiben (F2).

4.2.5 Richtlinien und Prozesse

Die Baloise hat Richtlinien für den Umgang mit OSS erarbeitet. Diese Richtlinien haben aktuell den Status einer finalisierten Guideline, welche sich nun bewähren muss. Es muss noch überprüft werden, ob diese Guideline tatsächlich zum Unternehmen passt und ob sie einen Mehrwert darstellt. In den Richtlinien sind keine Prozessschritte für das Nutzen von OSS und das Beitragen an anderen Projekten definiert (F2).

Das OS-Team hat ein *Template* erstellt, damit die Veröffentlichung auf *GitHub* vereinfacht wird. In diesem *Template* sind die Lizenzen *Contribution Guides und Readmes* enthalten (F2).

Für die Lizenzauswahl wurde mit der Rechtsabteilung zusammengearbeitet. Dabei wurden eine Vielzahl an Lizenzen geprüft und schliesslich eine Reihe von Standard-Lizenzen festgelegt, die sich für das *daily business* eignen, wie z.B. für die Freigabe von kleinen *Snippets* oder kleineren Projekten. Bei grösseren Projekten wäre eine weitere Zusammenarbeit mit der Rechtsabteilung nötig. Bei der Lizenzauswahl schaut die Baloise zuerst auf die Lizenzen der Ökosysteme. Wenn es keine gibt, wird die *Apache2* Lizenz empfohlen. Die Baloise verwendet auch *Scanning Tools*, um die Zusammenstellung der Lizenzen in einem Softwareprodukt analysieren zu können. Dabei handelt es sich aktuell noch um *proprietäre Tools, wie z.B. Nexus Lifecycle von Sonatype* (F2)

Für das Veröffentlichen von eigenen OSS-Komponenten wurden Prozessschritte definiert. Viele Projekte wurden erstmals nicht als OSS geschrieben, weshalb als erstes im Team entschieden werden muss, ob sich der Mehraufwand lohnt, bestimmte Komponenten als OSS zu veröffentlichen. Dafür muss mit den verschiedenen *Stakeholdern* im Team, dem *Product-* und *Application Owner* geklärt werden, ob sie bereit sind, den dabei entstehenden Mehraufwand aufzuwenden (F2).

Als nächstes findet eine Neutralisierung des Codes statt und eine Lizenz wird ausgewählt. Danach ist es wichtig eine Dokumentation zu erstellen, welche auch eine Beispielinstallation enthalten kann. Ebenfalls zum Prozess gehört, dass zwei Verantwortliche definiert werden für das Projekt, welches veröffentlicht werden soll. Diese zwei Verantwortlichen fungieren als Ansprechpartner für das betreffende Projekt. Externe Anfragen werden von den Projektverantwortlichen individuell beantwortet. Dafür besteht kein definierter Prozess (F2).

Vor der Veröffentlichung wird das OSS-Team für eine Überprüfung involviert. Dieses Team entscheidet dann zusammen mit dem restlichen Team darüber, was bei der bestehenden Version noch fehlt, welches die Gründe dafür sind und was noch verändert werden könnte. Ein offizielles Freigabegremium besteht aktuell noch nicht (F2).

In der Regel wird der gesamte Quellcode veröffentlicht mit dem Ziel, dass dieser einen Mehrwert stiftet. Der beste Zeitpunkt für eine Veröffentlichung ist nach eigener Aussage bereits ganz zu Beginn, bei der Planung und Ideenfindung. Dies ermöglicht es allen Beteiligten zu realisieren, dass es sich in diesem Fall um ein quelloffenes Projekt handelt. Je später dieser Schritt getan würde, desto mehr Arbeit würde es bedeuten, alle Aspekte transparent zu veröffentlichen (F2).

4.2.6 *Zusammenarbeit mit anderen*

Die Baloise betreibt keine eigene Community, ist aber in der *«Java User Group»* in der Schweiz, in *«CH Open»* und in der *«Eclipse Foundation»* aktiv. Wie oben erwähnt, wurde für das *Open Prevo* Projekt während sechs Wochen zusammengearbeitet. Dauerhaft wird aktuell jedoch mit anderen in OSS-Projekten zusammengearbeitet (F2).

Das Vorgehen der Baloise besteht darin, dass durch eigene Veröffentlichungen andere erstmals so einfach wie möglich die OSS-Lösungen der Baloise konsumieren können. Gelegentlich gibt es Beiträge in Form von Übersetzungen, *Buckreports* oder Fehlerbehebungen. Nach eigener Aussage kommen die Beiträge vor allem von privaten *GitHub* Accounts. Die Vermutung dahinter ist, dass im Unternehmen dieser Privatpersonen die Freigabe nicht geregelt ist. Falls die Beiträge von Externen zunehmen, ist eine *«Contributor Licence Agreements»* geplant, damit die Baloise das Ganze steuern und sich selbst schützen kann. Dies hängt damit zusammen, dass in den meisten Fällen der Arbeitgeber derjenigen Person, die etwas zurückmeldet, auch die Nutzungsrechte an diesem geistigen Eigentum hält. Der Nachteil einer solchen *«Contributor Licence Agreement»* ist, dass es Beitragende abschrecken kann, weil sie zuerst mit ihrem Arbeitgeber Abklärungen treffen müssen (F2).

Des Weiteren hat die Baloise bereits auch Dienstleister beauftragt, ein Modul oder kleinere Projekte mit einer Community zu schreiben und diese als OSS freizugeben. Nach eigener Aussage können durch solche Sponsorings OSS-Projekte

realisiert werden, ohne interne Ressourcen aufzuwenden, wie zum Beispiel Zeit für Mitarbeiterausbildung. Im Gegenzug erhält die Baloise die gewünschte OSS-Lösung und wird von der Community in Form einer Danksagung erwähnt (F2).

Die Vision von Baloise besteht darin, mit anderen Unternehmen OSS-Projekte in einer Industriearbeitsgruppe zusammen zu entwickeln und nicht nur eigene OSS-Projekte, welche durch sie getrieben ist, zu realisieren. Für eine solche Zusammenarbeit im grösseren Stil, braucht es gemäss den Erfahrungen des Gesprächspartners einen neutralen Grund, auf dem man sich bewegt. Aktuell verfolgt die Baloise mit anderen Versicherungen zusammen mit *«Open Insurance»* ein Projekt, welches ein herstellerneutrales Dach für alle OSS-Aktivitäten innerhalb der Versicherungsbranche zum Ziel hat. Dieses Projekt steht aktuell erst am Anfang und es gibt noch keine offizielle Arbeitsgruppe oder eine Rechtsform, sondern erst eine Website, einen *GitHub* Account und eine Reihe von Leuten, die sich zum Thema Open Insurance austauschen und Gedanken machen. Ziel ist es auch gemeinsam eine einheitliche Sicht auf die Nutzung und Freigabe von OSS zu schaffen. Eine Idee ist, dass man sich mit einer Non Profit OS-Organisation, wie zum Beispiel der *Eclipse Foundation* oder der *Linux Fundation* zusammentut, weil diese Erfahrungen mitbringen und einen neutralen Grund für eine unternehmensübergreifende Zusammenarbeit zur Verfügung stellen könnten. Die Baloise überlegt sich zurzeit, wo sie sich in diesem Prozess positionieren will. Dazu wurde das Business involviert und «Open Insurance» befindet sich zurzeit im Zustand eines *«Pre-Proposal»* (F2).

4.3 Helvetia Fallstudie

Die Helvetia Versicherung ist die führende Allbranchenversicherung der Schweiz. Zudem weist die Helvetia Versicherung profitable Marktpositionen in weiteren europäischen Ländern auf. Sie bietet sowohl für Privatpersonen als auch für Unternehmen zahlreiche verschiedene Versicherungen an: von privater und beruflicher Vorsorge über Haftpflicht-, Fahrzeug- sowie Transportversicherung bis hin zur Sachversicherung. Die Helvetia Versicherung unterstützt innovative Ideen und sieht in ihnen grosse Chancen für die Zukunft (Helvetia.com, 2019). So setzt die Helvetia Versicherung auch zahlreiche OSS-Lösungen ein, wie z.B. *Ansible oder Open Shift*. Die Helvetia Versicherung hat als Organisation keinen *GitHub* Account und hat bis anhin noch keine Projekte veröffentlicht (F3b). Das aktuelle Ziel von

Helvetia Mitarbeitenden ist nicht das Veröffentlichen von eigener ganzer Software, sondern das Beitragen zu anderen Open Source Lösungen, wie z.B. einzelne Komponenten der Infrastruktur (F3a).

4.3.1 Treiber für die Contribution in Open Source Projekten

Die Idee bei anderen OSS-Projekten beizutragen wird *Bottom-up* von den IT-Mitarbeitenden der Helvetia Versicherung getrieben. Die Mitarbeitenden der Helvetia Versicherungen setzten bereits viele OSS-Lösungen ein und wollten deshalb gerne etwas zurückgeben (F3b).

Das Management ist offen für neue Lösungen, hat selbst aber noch keine klaren Vorstellungen von der konkreten Umsetzung. Auch die Bedeutung von OSS für die Firma ist auf Managementebene aktuell noch nicht klar ausformuliert (F3a, F3b).

Aktuell kommet es bei der Helvetia Versicherung vor, dass Mitarbeitende mit ihrem privaten *GitHub* Account nach eigenem Ermessen etwas zu anderen Projekten beitragen. Das heisst, dass Mitarbeitende teilweise direkt an OSS mitarbeiten und *contributen*. Dies geschieht auch während der Arbeitszeit, stellt aber nicht unbedingt eine direkte Veröffentlichung von Codes durch die Firma dar. Die fehlende Regelung ist eine Herausforderung für die Mitarbeitenden. Aktuell ist nicht konkret definiert, wie die Rahmenbedingungen aussehen und was z.B. zurückgegeben werden soll bzw. darf (F3a). Mit OSS Kosten zu sparen, stellt kein Treiber für die Helvetia Versicherung dar. Es werden keine finanziellen Gründe genannt, sondern einerseits ideologische, andererseits praktische, da man sich davon eine Erleichterung der Arbeiten erhofft. (F3b).

4.3.2 Nutzen und Risiken in anderen Open Source Projekten beizutragen

Die OS-Lösungen, welche von der Helvetia Versicherung eingesetzt werden, sind zum Teil nicht optimal auf die eigenen Bedürfnisse zugeschnitten. So wird zwar generell vom Einsetzen von Open Source Lösungen profitiert, diese entsprechen jedoch teilweise noch nicht dem, was exakt nötig wäre (F3a). Durch *Contribution* in andere Open Source Projekte erhofft sich die Helvetia Versicherung eine aktivere Steuerung in der Entwicklung und eine schnellere Behebung von Fehlern(F3b).

Einen besseren Ruf in der IT Industrie zu bekommen, ist ein weiterer Nutzen, welcher im Gespräch erwähnt wird. Da man gerade als Finanzdienstleister nicht den

besten Ruf in der IT-Branche habe, wird gehofft, dass durch den Beitrag in Open Source Projekte, dieser Ruf verbessert werden kann (F3a).

Durch den Beitrag in Open Source Projekte möchte die Helvetia auch zu einem attraktiveren Arbeitgeber für Entwickler werden. Es wird vermutet, dass es aus Sicht von Entwicklern ein Plus darstellt, eine solche Strategie zu haben (F3b).

Ein weiterer Vorteil davon in anderen OSS-Projekten beizutragen ist, dass die Helvetia Versicherung die anfallenden Wartungsarbeiten so nicht selbst durchführen muss und auf diese Weise ein *Upstream* nehmen kann, ohne die eigenen Versionen warten zu müssen (F3a).

Ein Risiko durch den Einsatz von OSS-Lösungen und aktiver Beteiligung in den Communities sieht die Helvetia Versicherung in den zusätzlichen Personalkosten für den Aufbau von internem Know-how. Dies insbesondere, da ihr Kernbusiness nicht in der Softwareentwicklung liegt. So sei es nach eigener Aussagen bei Versicherungen vielmehr die Regel, dass kommerzieller Support angewendet würde und dafür die Personalkosten niedriger wären (F3a).

Weitere Risiken, welche jedoch als eher gering eingeschätzt werden, sind einerseits rechtliche Themen, in Form von Unklarheiten und Unsicherheiten bezüglich Lizenzfragen. Andererseits wird als potentielles Risiko ein Imageschaden genannt, der durch das Veröffentlichen von fehlerhaften Lösungen entstehen könnte. Die Imitationsgefahr stellt für die Helvetia Versicherung kein Risiko dar. Die im konkreten Fall eingesetzte Software wird nicht als entscheidender Faktor gesehen im Wettbewerb mit den anderen Versicherungen (F3a).

4.3.3 Strategische Bedeutung

OSS ist bei der Helvetia Versicherung strategisch noch nicht verankert. Doch in einigen Bereichen setzt die Helvetia Versicherung mittlerweile ausschliesslich auf OSS. Dies vor allem im Cloudbereich, jedoch mit kommerziellem Support (F3a, F3b). Dabei existieren jedoch keine Vorgaben, dass OSS bevorzugt werden sollte. Der Fokus liegt vielmehr immer auf dem konkreten Projekt. Ob dieses Projekt durch OSS umgesetzt wird, ist dabei zweitrangig (F3a).

Für den Betrieb von OSS-Lösungen wird vom Management der Helvetia Versicherung der kommerzielle Support von *Red Hat* bevorzugt, da *«strategisch eher in die Richtung gedacht wird, dass man die sogenannte Fertigungstiefe verringern möchte»* (F3a). Damit die Helvetia Versicherung die Systeme ohne kommerziellen

Support betreiben könnte, wäre ein grösseres Investment nötig. In Anbetracht dessen, wieviel die Helvetia für Support ausgibt und wieviel ein Mitarbeiter kostet, lässt sich relativ viel Support beziehen für das Geld, was für einen Mitarbeiter gezahlt würde. Da ein zusätzlicher Mitarbeiter diesbezüglich nicht reichen würde, wäre ein grösseres Investment nötig (F3a).

4.3.4 Interne Organisation

Aktuell gibt es bei der Helvetia Versicherung kein OSS-Team. Sie befindet sich diesbezüglich ganz am Anfang. So gehören die verschiedenen Beteiligten auch unterschiedlichen Ressorts an (F3a). Von den ca. 350 IT-Mitarbeitenden trägt nur ein kleiner Teil aktiv zu OS bei (ca. 5 bis 10 Personen). Ein paar IT-Mitarbeiter versuchen das Thema voranzutreiben, machen dies jedoch informell neben ihrer Tätigkeit und nicht, wie zum Beispiel bei der Baloise Versicherung, während der Arbeitszeit (F3a). Zukünftig ist kein Open Source Team geplant, sondern eine abteilungsübergreifende Zusammenarbeit. So sollen mehrere Leute aus verschiedenen Bereichen eine interne Community bilden und die Entwicklung treiben (F3b).

4.3.5 Aufbau von Richtlinien und Prozessen

Für den Einsatz von externen Softwarekomponenten gibt es bei der Helvetia Versicherung keine spezifischen Richtlinien. Der Einsatz dieser Softwarekomponenten wird durch kommerziellen Support geregelt. Der Aufbau von Richtlinien und Prozessen für das Beitragen in andere OSS-Projekten steht aktuell bei den IT-Mitarbeitern im Fokus. Dies, da der Einsatz von Community getriebene OSS in bestimmten Bereichen unerlässlich werden wird. Deshalb müsse geregelt werden, wie man mit solchen Lösungen umgehen soll, wie die Rahmenbedingungen aussehen und was auch zurückgegeben werden soll (F3a).

Die IT-Mitarbeiter, welche sich mit dem Thema OSS beschäftigen, machen aktuell Abklärungen, um Richtlinien zu definieren. Dabei sollen die gesetzlichen Voraussetzungen genauer betrachtet werden. Nach diesen Abklärungen besteht der nächste Schritt darin, dem Management den Mehrwert vom Beitragen in andere OSS-Projekte aufzuzeigen. In einem weiteren Schritt müssten weitere Entwickler ins Boot geholt, geschult und überzeugt werden. Insbesondere ist es dabei wichtig, dass die Entwickler ermutigt werden die neuen Lösungen vermehrt zu nutzen. Dafür soll den Entwicklern der Mehrwert der Lösungen mittels Beispielen aufgezeigt werden (F3b).

Der Prozess für das Beitragen an andere Projekten sollte so einfach wie möglich sein. Aus diesem Grund sollte dieser Prozess möglichst einfach gehalten werden: es sollen Richtlinien aufgestellt werden, damit die Entwickler sehen, welche Lizenzen sie nutzen dürfen und dass z.B. keine Passwörter oder sensible Daten veröffentlicht werden dürfen (F3b). Wie Entscheidungen bezüglich Veröffentlichungskomponenten getroffen werden sollen und wie der Freigabeprozess genau aussehen wird, ist aktuell jedoch noch unklar. Hierfür ist noch ein Gremium zu bestimmen (F3a).

Die Entscheidung für eine Veröffentlichung von Komponenten sollte frühzeitig getroffen werden. Es sollte von Anfang an klar sein, wenn etwas unter einer OSS-Lizenz veröffentlicht werden soll. Dies mit dem Ziel, den Aufwand zu sparen, der anfällt, wenn eine solche Entscheidung erst nachträglich getroffen wird (F3b).

Welche Lizenzen die Helvetia einsetzen möchte, ist ein offener Punkt und muss jeweils den Umständen entsprechend definiert werden (F3a). Beim Beitragen in anderen Projekten, ist es wichtig, dass die Lizenz der jeweiligen Projekte Freiheiten garantieren. Dies ist insbesondere wichtig, da die Helvetia keine eigenen Komponenten veröffentlicht, sondern vielmehr *contributet*. Unter diesen Bedingungen ist es zentral für sie, zu schauen, dass die Lizenzen keine Einschränkung darstellen und plötzlich die Pflicht besteht, den gesamten eigenen Code veröffentlichen zu müssen. Um dieses Risiko zu minimieren, ist eine Überprüfung mit einem Lizenz *Scanning Tool* geplant (F3b).

4.3.6 Zusammenarbeit mit anderen

Die Helvetia Versicherung arbeitet aktuell mit keiner Community zusammen. Es wurden, durch persönliche Kontakte, jedoch informelle Verbindungen aufgebaut. Dies z.B. mit der Baloise, der SIX Group, der Postfinance, der Mobiliar und Red Hat Von dieser informellen Zusammenarbeit erhofft sich die Helvetia Versicherung von den Erfahrungen der anderen Firmen lernen zu können für die Definition eigener Richtlinien. Dies passt für sie zur Idee von OSS, wonach man Sachen teilt, voneinander profitiert *«und es muss niemand das Rad neu erfinden»* (F3a).

Von der Zusammenarbeit mit einer Community erhofft sich die Helvetia Versicherung, dass Fehler schneller entdeckt und behoben werden und in einem weiteren Schritt zusätzliche Funktionalitäten mit der Community entwickelt werden. Die Helvetia Versicherung würde vorzugsweise einer bestehenden Community

beitreten. Vor allem wenn es um Produkte geht, die schon auf dem Markt sind, die also schon eine Community haben (F3a).

Indirekt hat die Helvetia Versicherung bereits mit einer Community zusammengearbeitet, indem sie ein Sponsoring gemacht hat. In diesem Sponsoring wurden Entwickler dafür bezahlt, gewisse *Features* zu implementieren. Diese *Features* werden nachher im OSS-Produkt verwendet. Der Nutzen dieser Vorgehensweise besteht für die Helvetia Versicherung darin, dass die Pflege der Softwarecodes gesichert ist und sie als Nicht-Softwareherstellerin sich dadurch nicht mehr vordergründig darum kümmern muss Die OSS-Community sieht jedoch nicht, dass die Helvetia Versicherung als Firma indirekt zu einem Produkt beiträgt. Es gibt bisher keine Produkte, wo der Name der Helvetia Versicherung draufsteht. Als Grund dafür wird die fehlende OSS-Strategie der Helvetia genannt (F3a).

4.4 SIX Group Fallstudie

Die SIX Group betreibt die Infrastruktur für den Schweizer Finanzmarkt und für einen breiten internationalen Kundenkreis. Sie bietet Produkte und Services für Wertschriftenhandel, Börsentransaktionen, Finanzinformationen und sichert den Informations- und Geldfluss zwischen Banken, Händler, Investoren und Dienstleistern. Das Unternehmen befindet sich im Besitz seiner Nutzer: rund 125 Banken (Six-group.com, 2019). SIX verwendet eine Vielzahl von OSS-Produkten, wie zum Beispiel *Akka* oder *Linux*, welche SIX helfen ihre Dienstleistungen effizient, sicher und stabil zu erbringen (Stürmer & Gauch, 2018, S.66). SIX hat keinen *GitHub* Account und *«aus Unternehmenssicht haben wir als SIX nichts veröffentlicht»* (F4b).

4.4.1 Treiber für die Veröffentlichung von Open Source Software

2016 spaltete sich bei der SIX die IT von der Business Unit ab, sodass alle IT Mitarbeiter in einer zentralen IT Abteilung vereint worden sind. Aufgrund dieser Zentralisierung wurde unter den Mitarbeitenden über Synergien, Erfahrungsaustausch und Gildensysteme nachgedacht und schliesslich eine Linux Gilde gegründet. Aktuell wird darin jedoch nicht nur auf Linux fokussiert. Die Gilde sollte zu einer allgemeinen OSS-Gilde werden. Das Vorgehen beim Aufbau der Gildensysteme war *Bottom-up* getrieben. Die Idee kam ursprünglich von einer Person und ist dann bei mehreren Mitarbeitenden auf Anklang gestossen. Anschliessend wurde diese Idee in einem sehr freien Rahmen umgesetzt (F4a). Die SIX betreibt als Organisation keinen *GitHub*

Account, aktuell kommt es aber vor, dass indirekt über den privaten *GitHub* Account OSS veröffentlicht wird. Das Ziel der OSS-Gilde ist es, eine Regelung für die Veröffentlichung von Codes zu entwickeln. (F4b).

4.4.2 Nutzen und Risiken eigene Projekte zu veröffentlichen

Einen grossen Nutzen, den die SIX in der Veröffentlichung insbesondere bei API sieht, sind die Erhöhung der Kundenzufriedenheit und die Akquisition von neuen Kunden. Dies z.B., indem eine Referenzimplementierung als OSS zur Verfügung gestellt würde. Die Vorgänge sollten kundenfreundlicher gestaltet werden, indem ein Kunde sich z.B. via einer fertigen *Client Library* auf das API *onboarden* kann. Gleichzeitig würde dies aber auch intern einen Nutzen stiften, indem die Referenzimplementierung beim *Testing* vom API eingesetzt werden könnte (F4a).

Ein weiterer Vorteil der Veröffentlichung liegt darin, auch bei kleineren Projekten aktiver steuern zu können. So z.B. beim *SaltStack Configuration Management*, welches intern verwendet wird. Bei diesem könnte Einfluss genommen werden, indem geregelt Codes veröffentlichen würden (F4b).

Ein weiterer Nutzen durch die Veröffentlichung von Projekten liegt darin, das eigene Image zu verbessern. SIX werde, nach eigener Aussage, aktuell als Amt für Datenverarbeitung wahrgenommen, was einen staubigen Eindruck hinterlassen würde. Dies stimme nicht mit den internen Entwicklungen überein. Die aktuellen Initiativen würden aus eigenem Interesse und nachhaltig verfolgt werden. Diese Botschaft sollte auch nach aussen vermittelt werden (F4b). Ein besseres Image würde auch die Arbeitgeberattraktivität steigern und mehr Talente anziehen. Es sei anspruchsvoll heutzutage die besten Entwickler zu rekrutieren. Im OSS-Bereich mit eigenen Projekten und Software präsenter zu sein, würde nach Einschätzung von SIX die Chancen erhöhen, kompetente Personen aus dem Entwicklerbereich anzuziehen (F4a).

Durch die Veröffentlichung von OSS könnte auch die Motivation bei den bestehenden Mitarbeitenden gefördert werden. Wenn diese wissen, dass der Code veröffentlicht wird, kann dies als Ansporn dienen, diesen möglichst gut zu entwickeln. Dies kann zu einer Qualitätssteigerung führen (F4b).

Ein weiterer Vorteil, der durch die Arbeit mit OSS-Lösungen entsteht, ist die Vielfalt an Lösungen für das gleiche Problem. Darunter kann dann die passendste Lösung im Einzelfall ausgesucht werden. Des Weiteren ist auch eine grössere Unabhängigkeit von den jeweiligen Marktleadern gegeben (F4b).

Auf der anderen Seite sieht die SIX jedoch auch Risiken in der Veröffentlichung von eigenen Projekten. Einerseits spielen dabei die entstehenden Kosten eine Rolle. Der Aufbau einer Community oder die öffentliche Betreibung eines Projektes werden als konkrete Kostenpunkte genannt. Früher hätte man noch geglaubt, dass mit OSS, Veröffentlichungen und Zusammenarbeit mit Communities vor allem Geld gespart werden könne. Dies wird heute differenzierter gesehen (F4a).

Ein weiteres Risiko stellt die Qualität der Veröffentlichungen dar, da bei Veröffentlichungen, die nicht hochqualitativ sind, ein Imageschaden drohe (F4a). Das Imitationsrisiko aufgrund von OSS-Veröffentlichungen wird als gering eingestuft. Ein solches Risiko in Form von Trittbrettfahrern und *Lost of intellectual property* wird eher in Bereichen gesehen, die weniger mit Technik zu tun haben (F4b).

4.4.3 Strategische Bedeutung

OSS ist bei der SIX strategisch nicht verankert. Die jetzige IT Strategie wird im Gespräch als *«sehr high-level»* bezeichnet und beinhaltet keine spezifischen Lösungen im Sinne von proprietärer – oder OSS. OSS strategisch zu verankern, stellt eine aktuelle Herausforderung dar (F4a).

Die Mitarbeitenden bei SIX sind aktuell im Austausch mit dem Management, welches eine offene Haltung gegenüber OSS hat und sowohl Interesse zeigt an OSS-, als auch an proprietären Lösungen. Die Begriffe, das Verständnis und das Ziel sind in Gesprächen mit dem Management klar definiert worden (F4b). OSS gewinne aktuell an Relevanz. Eine IT-Strategie für die Nutzung von OSS befindet sich in der Vernehmlassung- und Einsetzungsphase. Diese Strategie hält fest, dass im Plattformbereich OSS- gegenüber proprietären Lösungen Vorrang haben sollte, wenn diese gleichwertig sind (F4a). Ein Ziel der Gilde ist, dass auch das Beitragen an Projekten und das Veröffentlichen von OSS ein Teil der IT-Strategie wird (F4b).

4.4.4 Interne Organisation

SIX hat aktuell kein institutionalisiertes OSS-Team. Dies stellt insofern eine Herausforderung dar, dass verschiedene Themen in verschiedenen Teams aktuell sind und die Sichtbarkeit von teamübergreifenden Skills ungenügend ist (F4b). Wie oben bereits erwähnt existiert eine OSS-Gilde, auf welche im Folgenden genauer eingegangen wird. Das Ziel der Gilde besteht in der Förderung von teamübergreifender Zusammenarbeit (F4b). Aktuell gibt es zwei verschiedene Rollen in der OSS-Gilde;

einerseits den Head, andererseits die Members. Neu wird zusätzlich ein Sponsor aus dem Management zur OSS-Gilde hinzukommen. Weil dies erst seit zwei Monaten bestimmt ist, ist dieser im Moment noch nicht festgelegt (F4a).

Die Teilnahme an der OSS-Gilde ist freiwillig und umfasst ca. 50 Personen. Diese werden per E-Mail-Verteilerliste über aktuelle Aktivitäten informiert. Unter diesen Teilnehmenden beschäftigen sich fünf aktiv mit OSS-Themen (F4a).

Die Produktivität der SIX Mitarbeitenden wird durch die Rapportierung der Arbeitszeit gemessen. Um die freiwillige Teilnahme in der Gilde zu fördern, wird vom Management deshalb ein Anreiz gesetzt. Für die Gilde wurde ein offizielles Gefäss geschaffen. Wenn die Zeit für OSS-Themen im Sinne des Unternehmens genutzt wird, gilt diese Zeit als produktiv, was auch im persönlichen HR Performance Reporting auffällt. Die Zeit, welche ein Mitarbeiter für OSS-Themen aufwenden darf, ist jedoch begrenzt. Der maximale, nicht speziell zu genehmigende Zeitaufwand, beträgt 50 Stunden pro Jahr (F4a). Um OSS-Themen stärker voranzutreiben, müsste die OSS-Gilde mehr freiwillige Mitglieder anwerben. Eine Gilde mit vielen Mitgliedern würde es ermöglichen, verstreut über alle, viel Aufwand einzusetzen. Dies wird aktuell jedoch nicht als realistisch angesehen. Die meisten Aufgaben könnten nicht so parallelisiert werden, wie das in einem solchen Fall nötig wäre (F4a). Das *Mindset* müsse sich sowohl bei gewissen Mitarbeitern, als auch bei gewissen Managern noch ändern. Es wird gesagt, dass es aktuell sowohl Entwickler gäbe, die in OSS keinen Nutzen sehen, als auch Manager, welche dessen *Business Value* nicht sehen (F4a). Eine Vision besteht darin, dass die Mitarbeitenden mehr Freiheiten geniessen würden und einen Freiraum von 10% hätten, den sie selbstständig für Aktivitäten einsetzen könnten. Diese Vision sei zwar auf der Agenda, werde aktuell aus Ressourcengründen jedoch nicht gelebt (F4b).

4.4.5 Aufbau von Richtlinien und Prozesse

Es gibt bei SIX keine aktuelle Richtlinie für den Umgang mit OSS. Die bestehende Richtlinie sei alt und eher restriktiv ausgelegt (F4a). Die Unternehmensgrösse von SIX stellt für die OSS-Gilde eine Herausforderung dar bei der Erstellung von einheitlichen Richtlinien. Da es viele verschiedene Entwickler und somit viele Geschmacksrichtungen gibt, wird es als grosse Herausforderung erlebt, einen Konsens zu finden, der dann auch im Unternehmen umgesetzt werden kann (F4b).

Bevor Richtlinien für die Veröffentlichung von OSS gemacht werden können, muss zuerst einheitlich geregelt werden, wie externe Komponenten eingesetzt werden dürfen. Diese einheitlichen Richtlinien für die Nutzung existieren aktuell noch nicht (F4a). Eine solche Richtlinie für die Veröffentlichung zu erstellen ist ein längerfristiges Ziel der Gilde. Dies sollte bewirken, dass die Mitarbeitenden nicht mehr indirekt mit dem privaten Account veröffentlichen müssen und besser geschützt sind. So sollte diese aktuell bestehende Grauzone aufgelöst und geregelt werden. Ein weiteres Motiv für die Erstellung von Richtlinien besteht darin, nicht mehr über einen Anbieter wie *Red Hat* zu veröffentlichen. Als Beispiel wird die *Red Hat Enterprise* Linux oder OpenShift genannt, durch welche die SIX indirekt durch die Supportpreise an *Red Hat* etwas nach aussen gibt. Ziel ist es, dass die SIX nicht über einen solchen Anbieter veröffentlicht, sondern direkt als SIX in der Öffentlichkeit tätig ist(F4b). Der beste Zeitpunkt für eine Veröffentlichung eigener Projekte wäre gemäss den SIX Mitarbeitenden in der Entwicklungsphase, wo bereits ein Teil der Funktionalität sichtbar ist, im Sinne eines Prototyps (F4b).

Für den Aufbau von Prozessen plant die OSS-Gilde, den Rechtsdienst für einen Genehmigungsprozess zu involvieren. Konkret ist aber noch nicht klar, ob sich der Rechtsdienst ausschliesslich bei der Ausarbeitung von Richtlinien und Verhaltensanweisungen beteiligen würde, oder ob er auch in den jeweiligen konkreten Fällen involviert sein müsste. Lizenztypen für die Veröffentlichung wurden noch nicht bestimmt. Diese werden aktuell projektspezifisch ausgewählt. Sie können nicht generisch bestimmt werden, sondern müssen auf das spezifische Projekt und den jeweiligen Business Nutzen abgestimmt sein (F4a).

Zudem ist eine Mitarbeiterschulung geplant. Der Fokus liegt dabei auf den Lizenzthemen. Gemäss dem Feedback der Entwickler sind die groben Sachkenntnisse vorhanden. Was noch fehlt, ist eine Hilfestellung bei der Auswahl von Lizenzen, damit die Entwickler wissen, welche Lizenzen für welche Fälle geeignet sind (F4b).

4.4.6 Zusammenarbeit mit anderen

SIX arbeitet mit keiner OSS-Community zusammen und betreibt kein aktives Community Management. Der Fokus liegt auf der Ausarbeitung von Richtlinien für den Umgang mit OSS. Aus diesem Grund wurde Kontakt mit anderen Firmen aufgenommen, welche auf einer ähnlichen Entwicklungsstufe sind. Dadurch erhofft sich SIX einige Anhaltspunkte zu finden, wie solche Richtlinien aussehen könnten

(F4a). Von einer allfälligen Zusammenarbeit mit einer Community erhofft sich SIX zusätzliche Perspektiven, Fähigkeiten und Erfahrungen (F4a).

4.5 Amt für Informatik und Organisation des Kantons Bern Fallstudie

Das Amt für Informatik und Organisation (KAIO) ist die zentrale Ansprechstelle für die Behörden der Kantonsverwaltung Bern. Die Kantonsverwaltung besteht aus sieben Direktionen, der Staatskanzlei und der Justizverwaltung. Die sieben Direktionen bestehen aus insgesamt 44 Fachämtern, welche auf 270 Standorte verteilt sind. Das KAIO deckt die gesamte Grundversorgung der Informations- und Kommunikationstechnologie (ICT) mit einer breiten Palette an standardisierten Produkten und Dienstleistungen ab. Ziel von KAIO ist es neue technologische Lösungen auf eine effiziente, kostengünstige und leistungsfähige Art und Weise anzubieten (fin.be.ch, 2019).

Seit Mitte 2018 ist «Open Source Service» Teil der ICT-Grundversorgung von KAIO. Die Dienstleistung beinhaltet die Unterstützung der Fachämter und Direktionen bei der Identifikation geeigneter Applikationen, die veröffentlicht werden können. Zudem berät das KAIO die Fachämter bei der Veröffentlichung, der Lizenzierung der Applikation und der Betreuung der Community (Stürmer & Gauch, 2018, S.39). Seit Dezember 2018 hat der Kanton Bern einen *GitHub* Account, auf welchem die Direktionen und Ämter die Möglichkeit haben, eigene Anwendungen als OSS zu veröffentlichen. Im Dezember 2018 wurde die erste Anwendung *«ÖREB-Kataster Smart-Auszug»* der Bau-, Verkehrs- und Energiedirektion auf *GitHub* veröffentlicht. Mit dieser Anwendung können öffentlich-rechtliche Eigentumsbeschränkungen, wie z.B. Grundwasserschutzzonen oder Nutzungsplanung, auf einer Karte dargestellt werden (Finanzdirektion, 2018)

4.5.1 Treiber für die Veröffentlichung von Open Source Software

Die Idee, dass der Kanton Bern eigene Software veröffentlichen soll, entstand durch eine Motion im Grossen Rat und kann als Top-Down Ansatz gesehen werden, da der Grosse Rat als oberstes Organ gilt (F5). Ein Anliegen der Motion war, dass eigene Entwicklungen, bei welchen der Kanton Bern das Urheberrecht besitzt, als OSS freigegeben werden, wo dies sinnvoll erscheint (Motion 2013). Wie oben erwähnt, hat der Regierungsrat des Kantons Bern anfangs 2018 die neue Verordnung über die

Informations- und Telekommunikationstechnik der Verwaltung (ICTV) genehmigt. Mit der ICTV wurde unter anderem eine Rechtsgrundlage für die Publikation von OSS der Verwaltungen geschaffen (Vgl.2.5.2).

4.5.2 Nutzen eigene Projekte zu Veröffentlichen

Nützliche Aspekte, die das KAIO in der Veröffentlichung von eigenen Projekten sieht, sind die Förderung von Wettbewerb und das Ziel, unabhängiger gegenüber Lieferanten zu werden(F5).

Ein weiterer Nutzen für den Kanton besteht in der Schaffung von Transparenz. Diese Transparenz zu gewährleisten, wird nach dem Öffentlichkeitsprinzip als einer der wesentlichen Aufträge des Kantons angesehen. Jedoch werde diesem teilweise nicht ausreichend Rechnung getragen. Die Veröffentlichung von Projekten sollte dem entgegenwirken und Aussenstehenden ermöglichen zu sehen, was der Kanton macht und wo Teile der Steuergelder hinfliessen (F5).

Zudem erhofft sich das KAIO durch die Veröffentlichung von OSS, eine engere Kooperation zwischen den Behörden und Kantonen zu schaffen. Aktuell finde nur in seltenen Fällen eine interkantonale Zusammenarbeit statt. Dies soll sich durch die Veröffentlichung von Software ändern, indem dadurch mehr Kontakte zwischen den Kantonen geknüpft werden sollen und so gegenseitig voneinander profitiert werden kann.Von einer engeren Zusammenarbeit könnten alle Beteiligten profitieren, weil dadurch die Kosten aufgeteilt und verringert werden könnten (F5).

Vor allem bei Neuentwicklungen sieht das KAIO Sparpotential, da dort der finanzielle Zusatzaufwand kleiner ist, als wenn man eine proprietäre Software in eine OSS umwandeln will(F5). Deshalb besteht dort ein finanzielles Risiko, wo der Kanton bestehende Software veröffentlichen will, weil diese vor der Veröffentlichung angepasst werden muss. Als Beispiel wird eine Software der Steuerverwaltung genannt. Dort hätte eine Veröffentlichung zu hohen Kosten geführt (F5).

4.5.3 Strategische Bedeutung

Beim KAIO gibt es keine offizielle OSS-Strategie und OSS hat aktuell eine geringe strategische Bedeutung. Dass die Veröffentlichung von Software aktuell auf freiwilliger Basis geschieht, wird als Zeichen für die geringe strategische Bedeutung gesehen. Es wird erwartet, dass die strategische Relevanz zunehmen könnte, wenn die

Verbreitung von OSS zunehmen würde. Trotz fehlender Strategie wird OSS beim KAIO jedoch neuerdings in den Richtlinien als Alternative empfohlen (F5).

Das KAIO hat kein zentrales Budget, welches für die Veröffentlichung von OSS verwendet werden kann. Die finanziellen Kompetenzen wurden an die verschiedenen Ämter delegiert. Ein zentrales Budget für OSS würde dieser Strategie widersprechen, auch wenn es gleichzeitig motivierend wirken könnte (F5).

4.5.4 *Interne Organisation*

Beim KAIO beschäftigt sich aktuell eine Person mit OSS-Themen. Das KAIO betreibt selbst keine Systeme mehr, sondern stellt den Direktionen und Fachämtern einen Service zur Verfügung. Es ist auf übergeordneter Ebene tätig, in der Koordination von Lieferanten und Anspruchsträgern. Im Zusammenhang mit OSS bietet das KAIO den Direktionen eine Art Beratungsdienstleistung an. Diese beschränkt sich auf eine Initialberatung, wo der Service des KAIO vorgestellt wird. OSS-Schulungen für die Mitarbeitenden von KAIO, die Direktionen oder die Fachämter gibt es nicht. Solche sind auch nicht geplant und werden nicht als nötig angesehen, da nicht selbst publiziert wird (F5).

Seit der Veröffentlichung von «ÖREB-Kataster Smart-Auszug» hat sich bisher noch niemand gemeldet: «*Die Idee war auch, dass der eine oder andere sich meldet, aber das war bisher nicht der Fall»* (F5). Deshalb fokussiert sich das KAIO aktuell darauf, seinen Service bei den Direktionen und Fachämtern bekannter zu machen. Dazu sollen Kurzpräsentationen in den verschiedenen Gremien gehalten werden. Die Präsentationen werden zuerst bei den Geschäftsleitungen gehalten und sollen in einem zweiten Schritt auch auf tieferen Stufen gehalten werden. Dieser Ansatz wurde gewählt, weil die Softwareentwicklung und die Publikation von OSS für den Kanton Bern in erster Linie einen finanziellen Aufwand bedeutet und die Kosten von den einzelnen Direktionen und Fachämtern getragen werden. Aufgrund der bisherigen Erfahrungen sollen die Entscheidungsträger motiviert werden, einen gewissen Mehraufwand bei Neuentwicklungen zu tragen (F5).

4.5.5 Aufbau von Richtlinien und Prozesse

In der ersten Phase wurden auf juristischer Ebene die rechtlichen Rahmenbedingungen für die Veröffentlichung von OSS überprüft. Diese Überprüfung ergab, dass der Kanton keine Gesetzesänderung machen muss, um Software

publizieren zu dürfen. Im Anschluss wurden verschiedene Lizenzen analysiert und auf ihre Eignung überprüft. Dabei wurde die BSD Lizenz ausgewählt (F5).

Nach den juristischen Abklärungen wurden Hilfsmittel für die Direktionen und Fachämter erarbeitet und auf *GitHub* als Service publiziert. Bei den Hilfsmitteln handelt es sich vor allem um Checklisten und Rahmendokumente (F5).

Die Rahmendokumente beinhalten unter anderem Leitfäden für die Lizenzwahl oder die Community Gestaltung und Checklisten dafür, ob eine Software für eine Veröffentlichung geeignet ist und was bei der Dokumentation oder den Freigaben beachtet werden muss. Zudem gibt es zwei verschiedene Prozesse für die Veröffentlichung. Einerseits für die Veröffentlichung einer ganzen Software, anderseits für die Veröffentlichung von einzelnen Komponenten, wie einer *Library* ((GitHub Kanton Bern, 2019).

Das KAIO stellt eine Initialberatung und die oben erwähnten Dokumente zur Verfügung. Diese Dienstleistung in Anspruch zu nehmen, stellt eine Holschuld der interessierten Direktionen und Fachämter dar. Im weiteren Prozess der Entwicklung und Veröffentlichung ist das KAIO nicht mehr involviert. Danach sollen die interessierten Direktionen und Fachämter ihr Anliegen mit den jeweiligen Lieferanten zusammen bearbeiten und selber publizieren. Dabei erhält der Lieferant Zugriff auf den *GitHub* Account des Kantons Bern. Am Ende dieses Prozesses, nach der Publikation, würden dem KAIO die Publikation gemeldet. Auch die Entscheidung, ob eine Software oder nur Komponenten daraus publiziert werden, trifft nicht das KAIO. Das Amt, welches die Software entwickelt, ist auch für die Software verantwortlich. Dies wird als Fachverantwortung bezeichnet. Diese beinhaltet, dass das betreffende Amt darüber entscheidet, ob etwas publiziert werden darf (F5).

4.5.6 Zusammenarbeit mit anderen

Das Ziel von KAIO ist, dass die Direktionen und Ämter mit anderen zusammenarbeiten und OSS-Communities bilden, denn ohne Community wird es gemäss eigener Aussage schwierig von der Publikation langfristig zu profitieren. Für die Gestaltung einer Community wurde vom KAIO ein Leitfaden entwickelt, welcher dabei helfen sollte, mehr über die im Einzelfall passende Communityform herauszufinden. Für die Entscheidungsfindungen sind die Direktionen oder Ämter verantwortlich. Dabei hängt es davon ab, was publiziert und was damit erreicht werden sollte. Des Weiteren stellt sich die Frage, inwieweit die Kontrolle bewahrt werden und

inwieweit etwas frei gegeben werden sollte. Dies muss vom jeweiligen Fachamt im Einzelfall entschieden werden. Eine Community *Governance* gibt es zum aktuellen Zeitpunkt noch nicht (F5).

Wie oben erwähnt, wurde bisher eine OSS veröffentlicht, der «ÖREB-Kataster Smart-Auszug». Für die Entwicklung wurde eine externe Firma engagiert, welche das Projekt steuert. Die Publikation der Software, welche durch die Firma *Sturm und Bräm* entwickelt wurde, war von Anfang an geplant. Um diese Software herum ist eine inoffizielle Community entstanden. Die Kantone inklusive die Entwickler, welche an der Entwicklung mitgearbeitet haben, treffen sich regelmässig. Das Ziel dabei ist, die Software durch die inoffizielle Community weiter zu entwickeln. Die Treffen können als Ansporn dienen, einander mitzuziehen und weiterzuentwickeln. Bei zukünftiger Zusammenarbeit mit externen Lieferanten soll die Veröffentlichung von OSS in den Ausschreibungsprozess integriert werden. Es soll eine Bedingung sein, dass Software als OS publiziert werden darf. Diese Bedingung sollte dann von den Lieferanten als Muss-Kriterium erfüllt werden (F5).

4.6 Bundesamt für Landestopografie swisstopo

Das Bundesamt für Landestopografie swisstopo ist das Geoinformationszentrum der Schweiz. Swisstopo vermisst, erhebt und dokumentiert die Schweizer Landschaft, stellt geologische Daten und Karten oder Anwendungen wie den Kartenviewer des Bundes, *map.geo.admin.ch* im Internet zur Verfügung. Die Produktpalette ist breit (swisstopo.admin.ch, 2019a). Für die vorliegende Arbeit steht der Kartenviewer www.map.geo.admin.ch im Fokus. Geo.admin.ch wird von swisstopo betrieben. Die Daten und Frameworks zu geo.admin.ch sind auf GitHub veröffentlicht (Swiss Geoportal, 2019).

4.6.1 Treiber für Open Source basierte Lösung

Die Entwicklung und Veröffentlichung von OS basierten Lösungen wurde von den Mitarbeitenden von swisstopo initiiert. Der Prozess wird klar als *Bottom-up* und intrinsisch motiviert bezeichnet. Ein Treiber, das Geoportal OSS-basiert aufzubauen, war das Ziel, den Nutzern nicht nur Zugriff zu den Informationen und den Daten zu ermöglichen, sondern auch das Werkzeug, mit dem die Daten visualisiert werden können (F6).

Zudem gab es auf dem Markt keine Standardlösungen: *«Als wir es 2008 angefangen haben, gab es nichts COTS, also Commercial off-the-shelf, was wirklich performantes Webviewing von Geodaten zugelassen hätte»* (F6). Eine Option bestand darin eine solche Lösung zu beschaffen, wobei solche Beschaffungen beim Bund meist sehr lange dauern. Eine Alternative dazu bestand darin, sich nach einer OSS Community umzusehen. Dabei wurde ein Community basiertes OSS-Framework entdeckt. Dies deckte zu ungefähr 80 Prozent ab, was für die Umsetzung eines Viewers benötigt wird (F6).

Cloud Computing war ein weiterer Grund für die Wahl einer OSS basierten Lösung, da auf der Cloud nicht einfach *Dongles* vergeben werden können, um zu lizenzieren. Aus diesem Grund war OSS die naheliegendste Lösung, wenn horizontal skaliert werden sollte (F6).

4.6.2 Nutzen und Risiken durch Open Source basierte Lösung

Der Hauptnutzen von OSS-Lösungen liegt für swisstopo in der Unabhängigkeit und der Nachhaltigkeit. Es kann eine ganze Version mit internem Knowhow betrieben werden. So kann Abhängigkeit von Dritten vermieden werden. Zudem ist es nicht nötig durch ein Gremium zu gehen, wie dies bei kommerzieller Software nötig ist (F6). Diese Unabhängigkeit fördert auch die Motivation der Mitarbeitenden. Die Programmierer selbst meinen, dass es viel mehr Freude mache, an etwas zu arbeiten, was sie selbst beeinflussen und wo sie die Performance selbst machen können (F6).

Die Veröffentlichung von OSS fördert offene Schnittstellen, die Verbreitung der eigenen Lösung und ermöglicht eine weltweite Zusammenarbeit. Durch die Veröffentlichung und Zusammenarbeit mit der OSS-Community werden die aktive Einflussnahme und die schnellere Umsetzung von individuellen Bedürfnissen vereinfacht. Bei Standardlösungen müsse man häufig auf die nächste Version warten (F6).

Weil der Kartenviewer öffentlich ist und die Werkzeuge kostenlos genutzt werden können, ist es einfacher, Externe für die Nutzung der Geodaten zu sensibilisieren und zu schulen (F6).

Auf der anderen Seite gibt es auch gewisse Risiken, wie zum Beispiel die internen Kosten für die OSS basierte Lösung. Obwohl viele Leute dies meinten, sei OSS nicht kostengünstiger. Es sei schlussendlich genau gleich teuer wie kommerzielle Software. Das Geld würde anders investiert, z.B. in die Schulung von Mitarbeitenden

und zur Abdeckung anderer Risiken. Durch die Investition in Knowhow würde man das Risiko personeller Volatilität eingehen (F6).

Auch in der Zusammenarbeit mit der Community stellen sich gewisse Herausforderungen. Eine dieser Herausforderungen sei, dass die OSS-Community nicht in allen Ländern die gleiche legale Entität darstelle. Da es eine Community sei und kein legaler Körper, stelle das auch für den Bund eine Herausforderung dar (F6).

Eine weitere Schwierigkeit ist die personelle und themenspezifische Volatilität in der Community. Da die Community nicht stabil ist, kommt es häufig zu Themenwechsel, die im Voraus nicht absehbar sind (F6).

4.6.3 Strategische Bedeutung

Swisstopo verfügt über keine offizielle OSS-Strategie. OSS wird vielmehr als Methode angesehen denn als Ziel, welches strategisch verankert werden müsste. Die Anwendung von OSS ist *Bottom-up* getrieben und wird resultatorientiert eingesetzt, z.B. bei der Ausschreibung für ein Projekt (F6).

Dennoch ist die Bedeutung von OSS in der strategischen Stossrichtung 2020 im Zusammenhang mit *Open Government Data*-Grundsätzen wiederzufinden. Darin ist folgendes festgehalten: *«Swisstopo stellt seine amtlichen Daten und Produkte online kostenlos zur freien Wiederverwendung zur Verfügung»* (swisstopo.admin.ch, 2019b).

Gesetzlich unterliegt swisstopo dem Geo-Informationsgesetz. Dieses besagt, dass: *«sämtliche Geodaten allen – also Bund, Kanton und der Bevölkerung – für eine breite Nutzung, nachhaltig, rasch, einfach, in der erforderten Qualität und zu angemessenen Kosten, zur Verfügung stehen sollen (…) Das Ziel von geo.admin.ch ist die Umsetzung des Geoinformationsgesetzes»* (F6).

4.6.4 Interne Organisation

Bei swisstopo gibt es ein Team, welches sich hauptsächlich mit OSS-Projekten beschäftigt. Dabei handelt es sich um ungefähr zehn Personen, deren tägliche Arbeit von OSS dominiert ist. Es existiert ein «*DevOps* Team mit *continuous Integration*». Innerhalb des Teams gibt es verschiedene Rollen. Es gibt Entwickler, Projektleiter und Betreiber (F6).

Obwohl die Verwendung von OSS *Bottom Up* getrieben ist, ist eine Genehmigung durch eine übergeordnete Stelle notwendig. Als Bundesstelle wurde die

Entscheidung getroffen mit OSS zu arbeiten. Damit wurden auch die *share-alike* Lizenzbedingungen akzeptiert, die eindeutig besagen, dass Änderungen veröffentlich werden müssen (F6).

Entscheidungen auf der operativen Ebene werden im Team gefällt. Das Team wird aktiv einbezogen bei Entscheidungen, wo es z.B. darum geht nächste Generationen oder Versionen von Lösungen zu planen. Dabei entscheiden die Entwickler zusammen mit dem Projektleiter (F6).

Die Mitarbeitenden werden durch freiwillige und obligatorische Trainings weitergebildet. Diese Schulungen finden extern bei zertifizierten Unternehmen statt und werden im Rahmen von Weiterbildungen absolviert. Die Mitarbeiter stehen im Zentrum: *«Wir investieren mehr in Leute als in Software»* (F6). Dies wird von den Mitarbeitenden mit privatem Engagement zurückgezahlt, indem sie sich nebst den Schulungen häufig auch privat weiterbilden und in der Freizeit auf privater Basis in anderen OSS-Communities mitarbeiten (F6).

4.6.5 Richtlinien und Prozesse

Es gibt keine expliziten internen Richtlinien für den Umgang mit OSS bei swisstopo. Die Richtlinien für OSS stehen in den AGB des Bundes. Dort ist unter anderem festgehalten, wie OSS beschafft wird und wie man sie in den Betrieb übernimmt (F6). Für den militärischen Teil, welchen die swisstopo ebenfalls verwaltet, gibt es vom ISBO (Informatiksicherheitsbeauftragter Bund) eine Richtlinie. Vor allem, wenn es um die Integration von OSS in kritische Infrastrukturen geht, werden die Codes im Vorfeld durchgescannt (F6).

Swisstopo verwendet für die eigenen Projekte hauptsächlich die *BSD* oder *MIT* Lizenzen. Bei der Auswahl der OS-Lizenzen wird darauf geachtet, dass diese möglichst offen sind und Änderungen wieder zurückgeben werden müssen. In Fällen, wo es keine sonstige Alterative gibt, werden auch permissive OS-Lizenzen wie die *Apache* Lizenz verwendet, zum Beispiel bei 3D. Betreffend den Lizenzen wurde auch die Rechtsabteilung involviert, welche die OS-Lizenzen *gereviewed* hat und einen D*isclaimer* hinzugefügt hat (F6). Swisstopo scannt die Quellcodes auf GitHub durch und schaut ob ein Framework noch aktuell ist oder nicht. Sonst wird auch auf «*IA VIM»* entwickelt (F6). Vor der Veröffentlichung gibt es einen *Review* Prozess bei welchem das Vier-Augen-Prinzip angewendet wird. Ein Vorgesetzter führt es im Anschluss zusammen *(F6)*. Swisstopo veröffentlicht in der Regel den vollen

Quellcode, ausser es handelt sich um Themen, welche rein swisstopo- spezifisch oder von internen Projekten sind (F6).

Um möglichst erfolgreich zu sein, versucht swisstopo nach eigener Aussage sehr schnell und früh zu veröffentlichen. Dazu gehört auch die Veröffentlichung von nicht endgültigen Codes, welche noch nicht immer und überall funktionieren. Hinter diesem Vorgehen steckt die Überlegung, dass bei zu langen Wartezeiten das *«User Interface (UI)»* meist veraltet aussieht und nicht mehr zeitgerecht ist. Dies mindere die Akzeptanz bei den Kunden. Diese Herangehensweise stellt für den Bund eine grosse Herausforderung dar, da der Bund normalerweise nur Endgültiges veröffentlicht, wie ein Gesetz oder eine finale Landeskarte. Die neue Vorgehensweise, wonach auch Unvollendetes veröffentlicht wird, markiert einen Paradigmenwechsel und erfordert einen Wechsel des Mindsets (F6).

Auf externe Anfragen zu den Quellcodes reagiert die swisstopo proaktiv und fordert auf, diese zu benutzen. Dabei wird jedoch jede Garantie explizit abgelehnt. Falls jemand die Lizenzbestimmungen missachtet, schreibt swisstopos diejenige Person an. Eine solche Angelegenheit lasse sich meistens mit einem Email oder Telefon regeln (F6).

4.6.6 Zusammenarbeit mit anderen

Aus Ressourcengründe wird auch mit externen Auftragsnehmern zusammengearbeitet. Die Auswahl erfolgt dabei über Einladungsverfahren, Ausschreibungen, Konferenzen oder das bestehende Netzwerk (F6).

Unabhängig zu sein bleibt jedoch ein wichtiges Kriterium bei der Zusammenarbeit mit anderen. Das Ziel, die Lösung selbst zu betreiben und Fehler selbst zu lösen, bleibt bei der Zusammenarbeit ein wichtiges Ziel (F6).

Bei der Zusammenarbeit mit externen wird die Rechtsabteilung involviert. Dabei werden ein Werkvertrag und ein Forschungsvertrag aufgesetzt, welcher von der Rechtsabteilung überprüft wird. Diese Verträge beinhalten auch die AGBs vom Bund bezüglich OSS (F6).

Vor der Veröffentlichung der extern entwickelten Komponenten gibt es zudem einen Review Prozess, bei welchem ein interner Senior die OSS-Komponenten freigeben muss (F6).

Für swisstopo ist die Zusammenarbeit mit OSS-Communities wichtig. Es wird aktiv mit der *«Openlayers Community»* zusammengearbeitet. In weiteren OSS-

Communities wie z.B. «Open Geospatial Consortium» oder «Angular» nimmt die swisstopo eine beobachtende Rolle ein.Teilnehmer der Communities sind Unternehmer, aber auch private Entwickler. Die OSS-Communities sind sehr international, doch es gibt auch in der Schweiz OSS-Communities. Beim Beitritt in eine Community wird darauf geachtet, dass diese genügend gross ist, dass sie also eine kritische Grösse aufweist. Des Weiteren soll sie nachhaltig, offen und zukunftsträchtig sein. Zudem hat swisstopo eine eigene OSS-Community aufgebaut; *api3.geo.admin.ch*. Darin sind ungefähr 400 Personen vertreten. Wenn ein Problem im Framework auffällt, wird dieses dort reingeschrieben (F6).

Das Vorgehen beim Aufbau folgte dem *Trial-and-Error*-Prinzip. Konkret wurden Leute angeschrieben und angerufen, es wurden E-Mail-Listen aus Projekten erstellt, wie Schulen, Fachhochschulen oder Konferenzen in der ganzen Schweiz. Die Community ist organisch und aus intrinsischer Motivation heraus gewachsen. Des Weiteren ist swisstopo auch in den sozialen Medien aktiv. So wurde z.B. eine Twitter Community, zusammen mit dem MS Geo Webforum, aufgebaut (F6).

Swisstopo betreibt ein aktives Community Management. Es wurden verschiedene Foren aufgebaut, wie z.B. das API3 Google Groups Forum. Zudem werden Fehlermeldungen über *GitHub* kommuniziert. Private werden ermutigt zu *contributen*, wie z.B. Fehler zu beheben. Zusätzlich ermutigt swisstopo auch andere in eine Weiterentwicklung zu investieren und macht aktuell zum Beispiel ein Crowdfunding für Openlayer 6. Innerhalb von drei Wochen wurde 200 000 USD gespendet. Zu den Spendern gehören Private, Kantone oder multinationale Unternehmen, welche Interessen an einer Weiterentwicklung haben. Dabei spiele die zwischenmenschliche Kommunikation eine entscheidende Rolle, da es darum geht, andere vom Potential von OSS-Lösungen zu überzeugen (F6).

Der Beitrag von swisstopo an die Community ist einerseits finanziell, anderseits wird mitgeholfen, Standards zu entwickeln und Risikoabschätzungen vorzunehmen. Zudem werden Kompetenzen aufgebaut und Schulungen veranstaltet und koordiniert. Andererseits erwartet swisstopo von der Community eine gewisse Offenheit und die Bereitschaft zu investieren, auch mit *Manpower*. Darüber hinaus wird erwartet, dass die Community intrinsisch motiviert ist (F6).

Die Entscheidungsfindung innerhalb der Community findet eher zentral statt. Es wird angemerkt, dass dies zwar zu einem gewissen Grade dem Community

Gedanken widerspreche. Es verhalte sich aber häufig so, dass diejenige Person, die bezahlt, auch bestimme. Trotzdem versucht swisstopo eine Balance zwischen Offenheit und Kontrolle zu finden. Es sei wichtig die Community zusammenzuhalten. Dafür müsse man eine gewisse Offenheit zeigen. Ansonsten lande man wieder im Bereich der kommerziellen Lösungen. Wenn Fehler gefunden würden, würden diese beseitigt. Es wird von einer roten Linie gesprochen, die es nicht zu überschreiten gelte, wie z.B. das Veröffentlichen von *Username* oder Passwörtern (F6).

5 Fallübergreifende Interpretation der Ergebnisse

Im Folgenden werden die wichtigsten Ergebnisse anhand einer fallübergreifenden Analyse zusammengefasst. Dabei werden die Daten auf unterschiedliche Arten betrachtet und interpretiert (Eisenhardt, 1989, S540). Als erstes werden fallübergreifende Nutzen und Risiken analysiert. Danach werden verschiedene Arten der Veröffentlichung, welche in den verschiedenen Interviews vorgestellt wurden, präsentiert. Als drittes werden die Treiber für die Veröffentlichung von OSS erläutert und die Herausforderungen der verschiedenen Treiber analysiert. Zum Schluss werden verschiedene Formen der Zusammenarbeit, welche aus den Interviews hervorgingen, vorgestellt.

5.1 Nutzen und Risiken bei der Veröffentlichung von OSS

Zusammenfassend lässt sich feststellen, dass der wahrgenommene Nutzen durch die Veröffentlichung von eigenen Codes bei den befragten Unternehmen und Behörden gegenüber den wahrgenommenen Risiken überwiegt. Die Abbildungen 5 und 6 visualisieren die Ergebnisse und umfassen je fünf Nutzen und Risiken, die herausgebildet wurden und in den folgenden Abschnitten beschrieben werden.

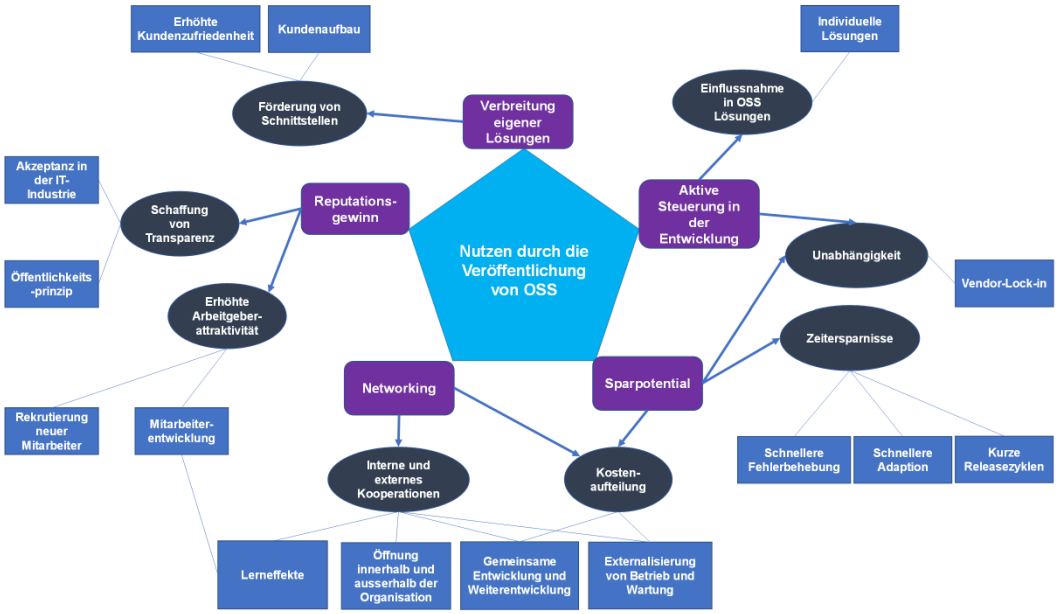

Abbildung 4: Nutzen bei der Veröffentlichung von OSS

5.1.1 Nutzen bei der Veröffentlichung von OSS

Indem Unternehmen und Behörden OSS veröffentlichen, erhöht sich ihre Sichtbarkeit sowohl innerhalb als auch ausserhalb der Organisation. Die Fallstudien

haben gezeigt, dass durch die Thematik OSS interne *«Silos»* aufgebrochen werden können und teamübergreifende Zusammenarbeit gefördert werden kann. Auch führt die Veröffentlichung dazu, dass neue Kontakte mit externen Entwicklern, Unternehmen, Behörden und Organisationen geknüpft werden können. Wenn Unternehmen und Behörden eigene Lösungen veröffentlichen und ihnen bekannt ist, welche externen Organisationen oder Entwickler ihre OSS-Lösung verwenden, können sie diese für Weiter- oder Neuentwicklungen proaktiv ins Boot holen und somit anfallende Kosten und Arbeitsaufwand aufteilen. Neben der Kostenersparnis generiert die Kooperation auch einen Lerneffekt für die eigenen Mitarbeitenden, weil sie z.B. neue Entwicklungsmodelle von erfahrenen externen Entwicklern kennenlernen und sich dadurch weiterentwickeln können.

Sparpotential besteht auch in zeitlicher Hinsicht. In diesem Zusammenhang wurde von den befragten Unternehmen und Behörden auch die Unabhängigkeit gegenüber den grossen Softwareherstellern erwähnt, da ihre Wünsche für Fehlerbehebungen oder Funktionsverbesserung nicht mit der Agenda der grossen Softwarehersteller übereinstimmen müssen. Dadurch können Fehler schneller behoben und individuelle Anpassungen vorgenommen werden. Zudem stehen die angepassten Versionen so schneller zur Verfügung und der notwendige Gang durch ein Gremium entfällt, wie dies bei kommerzieller Software der Fall ist.

Des Weiteren kann das Veröffentlichen von OSS-Lösungen für Unternehmen oder Behörden aus Reputationsgründen Sinn ergeben. Einerseits erhoffen sich die befragten Unternehmen und Behörden eine erhöhte Arbeitgeberattraktivität, sowohl bei der Rekrutierung von neuen als auch für die bereits bestehenden Mitarbeiter. Die Unternehmen und Behörden werden als moderner wahrgenommen und die Entwickler erhoffen sich in einem solchen Umfeld mehr individuelle Einflussmöglichkeiten und mehr Chancen auf eigene Weiterentwicklung, als dies in einem Umfeld der Fall ist, in welchem auf Standardsoftware von grossen Unternehmen gesetzt wird und so weniger Freiheiten möglich sind.

Anderseits schaffen Unternehmen und Behörden durch die Veröffentlichung auch Transparenz, welches für das Vertrauensverhältnis förderlich ist. Bei den Behörden oder Medien spielt das Öffentlichkeitsprinzip eine entscheidende Rolle. Das Veröffentlichen wirkt unterstützend beim Schaffen von Transparenz, indem die Bürger sehen wie öffentliche Gelder eingesetzt oder wie Daten verarbeitet werden. Bei

den befragten Unternehmen hingegen geht es in erster Linie um die Akzeptanz in der IT-Industrie, da sie als Versicherung oder Finanzdienstleister dort nicht den besten Ruf geniessen.

Ein weiterer Vorteil der Veröffentlichung ist die aktive Steuerung bei der Entwicklung von kleineren OSS-Projekten. Oft entsprechen die verfügbaren Lösungen nicht vollumfänglich den Bedürfnissen der befragten Unternehmen und Behörden. Indem sie eigene Funktionsverbesserungen veröffentlichen, können sie vermehrt eigene Wünsche für die Weiterentwicklung einer OSS-Lösung einbringen. So kann es sein, dass jemand anderes die gewünschte Lösung bereits besitzt oder die Dringlichkeit eine Lösung zu finden höher ist. So kann gegenseitig voneinander profitiert werden.

Die Veröffentlichung von OSS dient Unternehmen und Behörden vor allem bei der Verbreitung eigener Lösungen, insbesondere wenn sie eine eigene Schnittstelle fördern wollen. Durch die Veröffentlichung kann die Kundenzufriedenheit erhöht werden, indem z.B. eine Referenzimplementierung mitveröffentlicht wird. Zusätzlich kann die Veröffentlichung auch der Kundenakquisition dienen.

5.1.2 Risiken bei der Veröffentlichung von OSS

Die Veröffentlichung von OSS bringt jedoch nicht ausschliesslich, wie oben erwähnt, einen Reputationsgewinn, sondern es besteht auch die Gefahr, dass ein Unternehmen oder eine Behörde dadurch einen Reputationsverlust erleidet. Dies kann geschehen, wenn eine instabile und fehlerhafte Lösung im Namen des Unternehmens oder der Behörde veröffentlicht wird.

Abbildung 5: Risiken bei der Veröffentlichung von OSS

Des Weiteren wurden die durch die Veröffentlichung von OSS anfallenden Kosten unterschiedlich bewertet. Aus den Gesprächen ging die Vision hervor, dass langfristig durch das Veröffentlichen von OSS Kosten gespart werden können. Dies wird auch als Argument gegenüber Entscheidungsträgern verwendet. Jedoch wurde auch darauf hingewiesen, dass OSS-Lösungen nicht immer günstiger sind. Vor allem wenn eine proprietäre Lösung in eine OSS-Lösung umgewandelt werden soll, besteht die Gefahr, dass der Mehraufwand für die Umwandlung und die Veröffentlichung den Nutzen für das Unternehmen oder die Behörde übersteigt. Zudem müssen für die Pflege von OSS-Lösungen interne Ressourcen in Form von Know-how aufgebaut werden, welche bei den meisten befragten Unternehmen und Behörden intern aktuell nicht vorhanden sind. Deshalb wird eine proprietäre Lösung oder eine OSS-Lösung mit kommerziellem Support von einem Teil der Befragten bevorzugt.

Ein weiteres Risiko, welches im Zusammenhang mit der Veröffentlichung und Investition in OSS-Lösungen genannt wurde, ist die Volatilität. Einerseits gibt es intern eine gewisse Mitarbeiterfluktuation, durch welche Wissen verloren geht, anderseits kommt es auch bei der externen Zusammenarbeit zu Veränderungen, wenn z.B. unerwartet der Hauptentwickler einer Community verschwindet.

Im Zusammenhang mit der Community besteht zudem ein Haftungsrisiko, weil die OSS-Communities keine legale Entität darstellen und sie deshalb z.B. nicht verklagt werden können, wenn ein gewünschtes Ergebnis nicht zustande kommt.

Ein weiteres Risiko stellen rechtlichen Aspekte dar, insbesondere die Anwendung von OSS-Lizenzen. Die befragten Unternehmen und Behörden sehen ein Risiko darin, dass die Mitarbeitenden bei der Lizenzauswahl für die Veröffentlichung von OSS zu wenig sensibilisiert sind.

Zuletzt wird auch das Risiko genannt, dass die Veröffentlichung von OSS zu einem Wettbewerbsverlust führen kann, insbesondere wenn es sich um ein strategisch wichtiges Projekt handelt.

5.2 Direkte und indirekte Veröffentlichung

Durch die Gespräche mit den neuen Akteuren konnten unterschiedliche Arten der Veröffentlichung von eigener Software und Codes herausgebildet werden. Dabei kann zwischen direkter und indirekter Veröffentlichung unterschieden werden, welche in Abbildung 7 visualisiert und im folgendem genauer erläutert werden.

Eine indirekte Veröffentlichung liegt vor, wenn OSS nicht im Namen des Unternehmens veröffentlicht wird. Sie kann zum einen durch Mitarbeitende erfolgen, welche während der Arbeitszeit mit ihrem privaten *GitHub* Account Fehler an die Community melden, oder Funktionsverbesserungen und Codes veröffentlichen. Die Mitarbeitenden sind bei diesem Vorgehen nicht geschützt. In diesem Fall ist das Unternehmen oder die Behörde für die Öffentlichkeit nicht sichtbar und profitiert nicht von dem oben erwähnten Nutzen, welcher durch die Veröffentlichung von OSS generiert werden kann.

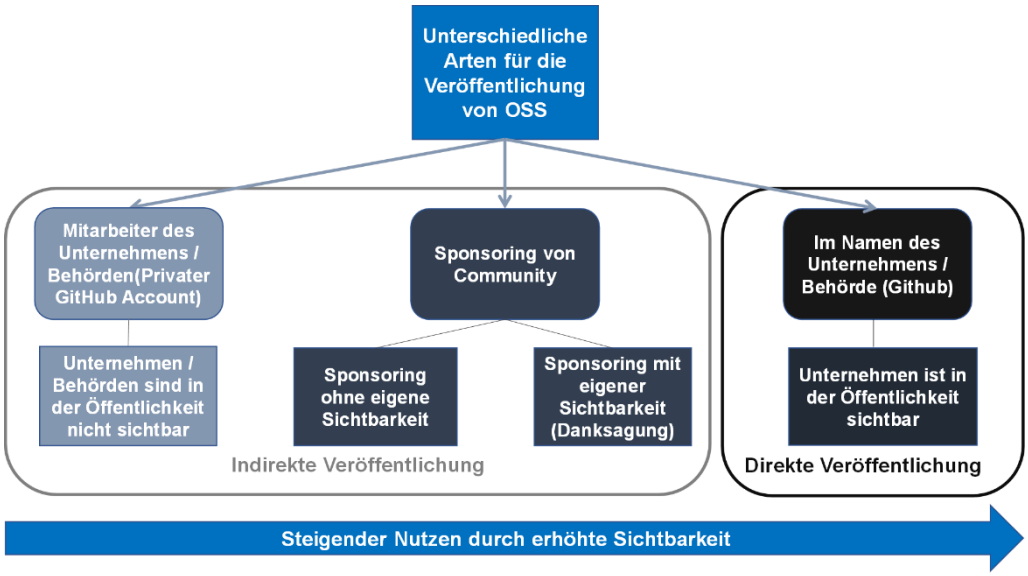

Abbildung 6: Unterschiedliche Arten OSS für die Veröffentlichung von OSS

Zum anderen können Unternehmen oder Behörden auch durch Sponsoring indirekt OSS veröffentlichen. In einem solchen Fall wird ein Entwickler oder eine Community für die Entwicklung bezahlt. Die Lösung wird unter dem Namen des Entwicklers oder der Community als OSS veröffentlicht. Bei der Veröffentlichung gibt es die Optionen, dass der Sponsor gar nicht erwähnt wird oder aber eine Attribution in Form einer Danksagung an den Sponsor getätigt wird. Durch die Erwähnung wird der Sponsor für die Öffentlichkeit sichtbar und hat dadurch einen gewissen Nutzen durch die Veröffentlichung von OSS, wie z.B. einen steigenden Reputationsgewinn oder die Akzeptanz in der Community.

Eine direkte Veröffentlichung liegt vor, wenn einzelne Codes oder eigene OSS-Projekte im Namen eines Unternehmens oder einer Behörde veröffentlicht werden. Die befragten Akteure betreiben dabei einen *GitHub* Account als Unternehmen oder Behörde und sind somit für die Öffentlichkeit sichtbar. Eine

erhöhte Sichtbarkeit der neuen Akteure führt auch dazu, dass der volle Nutzen durch die Veröffentlichung von OSS ausgeschöpft werden kann.

5.3 Treiber und Herausforderungen bei der Veröffentlichung von OSS

In dieser Studie konnten zwei verschiedene Treiber für das Veröffentlichen von OSS bei Unternehmen und Behörden in der Schweiz herauskristallisiert werden. Bei fünf befragten Unternehmen und Behörden handelt es sich um eine *Bottom-up* Initiative, da einzelne Mitarbeitende sich in ihrem Unternehmen oder ihrer Behörde dafür einsetzten, dass diese OSS veröffentlichen kann. Eine Ausnahme unter den Befragten bildet das KAIO, bei welchem man von einem *Top-down* Ansatz sprechen kann, weil der Auslöser eine Motion im Nationalrat war. Im Folgenden werden die Herausforderungen beider Ansätze genauer betrachtet.

5.3.1 Herausforderungen bei einem Top-Down Ansatz

Im Fall von KAIO wurde die Motion, die vom Grossen Rat einstimmig beschlossen wurde, umgesetzt. Der Unterschied zu der *Bottom-up* Initiative ist, dass die Initianten bei der Umsetzung nicht involviert sind, sondern das KAIO als zentrale Ansprechstelle für Informatik und Organisation für die Behörden der Kantonsverwaltung Bern damit beauftragt wurde. Das KAIO hat im Anschluss mithilfe einer Pilotapplikation, dem «ÖREB-Kataster Smart-Auszug», Grundlagen in Form von Richtlinien, Checklisten und Prozessen geschaffen. Dies mit dem Ziel, dass weitere Ämter und Direktionen des Kantons ihre Softwarelösungen oder Teile davon veröffentlichen können. Die Herausforderung besteht darin, weitere Ämter und Direktionen zu überzeugen diesen Weg zu gehen. Obwohl gesetzliche Grundlagen geschaffen und eine OSS-Dienstleistung zur Verfügung gestellt wurden, zeigt dieser Fall auf, dass die Etablierung einer OSS-Kultur nicht einfach von oben nach unten festgelegt werden kann. Damit weitere Ämter und Direktionen des Kantons mitziehen, scheint es wichtig, internes Marketing unter den verschiedenen Ämtern und Direktionen zu betreiben. Dabei scheint es sinnvoll dieses sowohl auf operativer -, als auch auf Führungsebene vorzunehmen.

Dieser Fall zeigt auch auf, dass die Überzeugungsarbeit grösser ausfällt, wenn die Kosten, welche im Zusammenhang mit OSS-Publikation entstehen, nicht von der

beratenden Stelle (KAIO) getragen werden, sondern von den zu überzeugenden Ämtern und Direktionen getragen werden müssten.

5.3.2 Herausforderungen bei einem Bottom-Up Ansatz

Bei den *Bottom-Up* Ansätzen ist eine der grössten Herausforderungen, das eigene Umfeld von der Veröffentlichung von OSS zu überzeugen. Bei einem *Bottom-Up* Ansatz geht die Idee von einzelnen Mitarbeitenden aus. Je grösser die Unternehmen, die Behörden und die involvierten Teams sind, desto mehr Überzeugungsarbeit, Koordination und Schulung werden notwendig. Bei den grösseren Unternehmen wurde zudem erwähnt, dass sich das *Mindset,* vor allem von älteren Mitarbeitenden, ändern müsse, um neue Wege gehen zu können. Auch müssen Richtlinien erarbeitet werden, die definieren, wie die Mitarbeitenden im jeweiligen Unternehmen oder der jeweiligen Behörde OSS veröffentlichen sollen. Weil die initiierenden Mitarbeitenden keine Entscheidungsgewalt innehaben, müssen sie sicherstellen, dass die Geschmacksrichtungen aller Beteiligter einbezogen werden, damit alle an einem Strang ziehen und die Unternehmen und Behörden einheitlich nach aussen auftreten durch die Publikation.

Das Management davon zu überzeugen, stellt bei den Befragten eine geringere Herausforderung dar als die Überzeugungsarbeit, welche bei den Mitarbeitern geleistet werden muss. Obwohl das generelle Verständnis beim Management vorhanden ist, wurde die Veröffentlichung von OSS bei keinem der befragten neuen Akteure strategisch offiziell verankert. Das Management stellt insofern eine Herausforderung dar, weil es bestimmt wie das vorhandene Geld investiert wird.

5.3.3 Wie das Geld für OSS-Lösungen investiert werden kann

Ein Grund für die fehlende strategische Verankerung ist, dass sich alle befragten Unternehmen und Behörden erst am Anfang des Prozesses befinden und sich dieser noch etablieren muss. Ein weiterer Grund liegt darin, dass die Softwareentwicklung nicht zum Kerngeschäft der Befragten gehört und OSS eher als Methode zur Zielerreichung angesehen wird. Wenn der Bedarf nach einer OSS-Lösung auftaucht, wurden drei verschiedene Strategien genannt, wie das Geld für eine OSS-Lösung investiert werden kann. Die drei verschiedenen Strategien sind in Abbildungen 8 visualisiert und werden im Folgenden noch genauer erläutert.

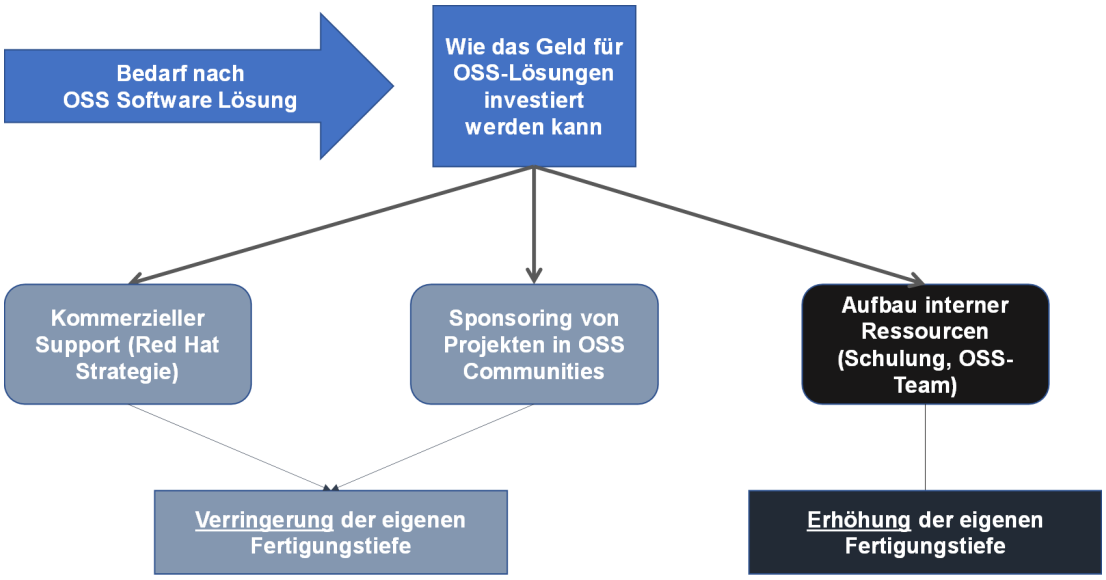

Abbildung 7: Wie das Geld für OSS-Lösungen investiert werden kann

Eine Strategie liegt darin, die Pflege einer OSS-Lösung durch einen kommerziellen Support zu sichern. Vorteil dabei ist, dass die Fertigungstiefe verringert werden kann und keine internen Ressourcen aufgebaut werden müssen. Nachteil ist, dass dabei nicht im Namen des Unternehmens oder der Behörde OSS veröffentlicht werden kann, sondern dies indirekt über die jährlichen Supportpreise geschieht, welche an einen OSS-Dienstleister, wie z.B. RedHat gezahlt wird. Der Nutzen der Veröffentlichung kommt so nur dem OSS-Dienstleister zu und nicht dem Unternehmen. Auch wenn Unternehmen den OSS-Dienstleister - im Gegensatz zu einer proprietären Lösung - einfacher wechseln können, kann man in diesem Zusammenhang von einer Abhängigkeit von einem OSS-Dienstleiter sprechen, welche man bei dieser Strategie in Kauf nimmt.

Eine zweite Strategie ist, wie oben bereits erwähnt, das Sponsoring von OSS Communities, um eine OSS-Lösung zu erhalten. Der Vorteil besteht auch hier darin, dass keine zusätzlichen internen Ressourcen aufgebaut werden müssen und die Fertigungstiefe verringert werden kann. Nachteil ist, dass man als Unternehmen gar nicht oder nur bedingt durch eine Attribution sichtbar wird. Somit ist der Nutzen durch die Veröffentlichung der initiierten OSS-Lösung sehr gering.

Der Aufbau von internen Ressourcen, um eigene OSS-Projekte zu starten, ist die dritte Strategie, welche genannt wurde. OSS-Lösungen werden durch interne Mitarbeitende entwickelt, betreut und veröffentlicht. Die Mitarbeitenden werden aktiv geschult und beschäftigen sich hauptsächlich mit OSS. Dadurch kann das

Unternehmen den vollen Nutzen ausschöpfen, der durch die Veröffentlichung von OSS entsteht, weil es für die Öffentlichkeit sichtbar wird (vgl. 5.1.1). Der Nachteil dieser Strategie liegt darin, dass die Fertigungstiefe erhöht wird, weil interne Ressourcen für ein OSS-Team verwendet werden und dieses geschult werden muss.

5.4 Zusammenarbeit mit anderen

In den Fallstudien wurden drei verschiedene Herangehensweisen genannt, wie Unternehmen und Behörden mit anderen im Kontext von OSS zusammenarbeiten. Diese Herangehensweisen werden in Abbildung 9 abgebildet und im Folgenden genauer erläutert.

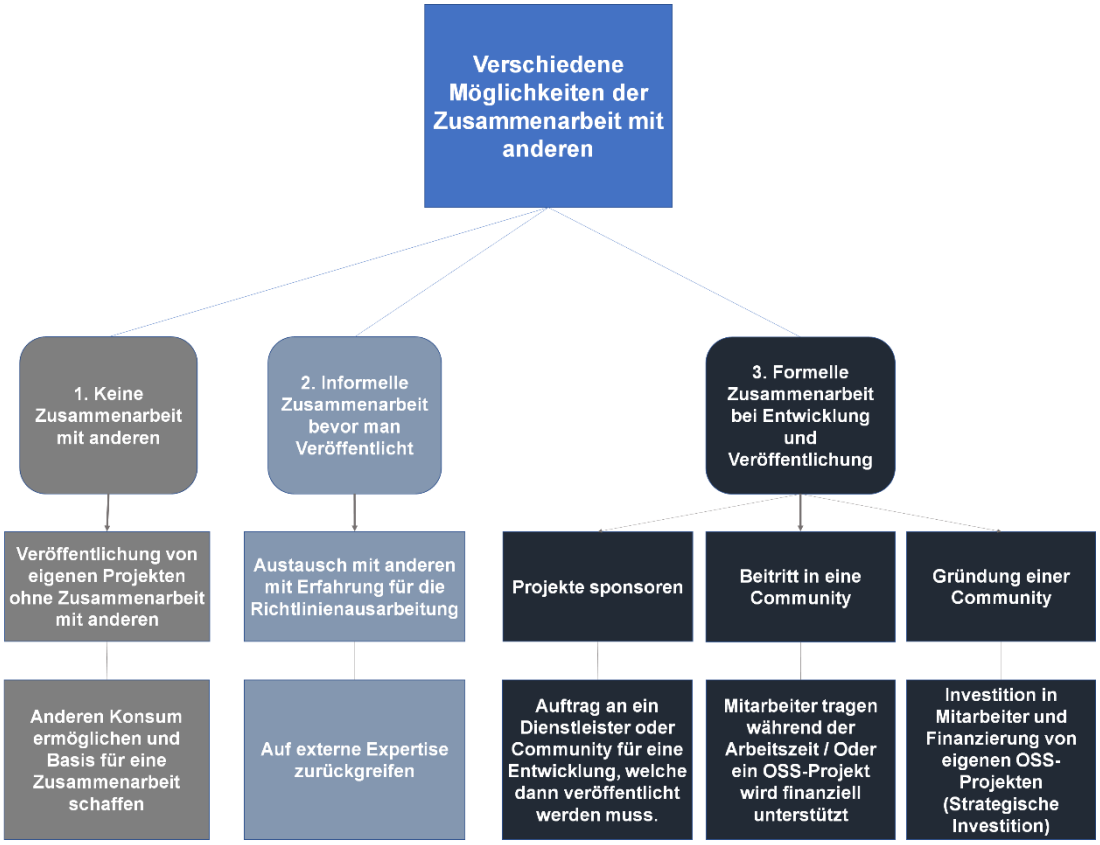

Abbildung 8:Verschiedene Möglichkeiten der Zusammenarbeit mit anderen

Eine Herangehensweise besteht darin, dass die Veröffentlichung von OSS oder Codes an sich einen Mehrwert bietet, indem anderen der Konsum ermöglicht wird und auf dieser Arbeit aufgebaut werden kann oder ein wichtiges Ziel in der Schaffung von Transparenz liegt. Dabei wird keine unmittelbare Zusammenarbeit mit anderen angestrebt.

Eine zweite Möglichkeit liegt darin, mit anderen zusammenzuarbeiten, um Erfahrungen auszutauschen, damit Richtlinien erarbeitet werden können. Diese Vorgehensweise wurde von Unternehmen genannt, welche bis anhin noch nicht im Namen des Unternehmens veröffentlicht haben, wo unter den Mitarbeitenden jedoch der Wunsch besteht, dies in Zukunft zu tun. Interessant hierbei ist, dass der OSS-Gedanke nicht nur für das Teilen von Codes gelebt wird, sondern auch für das Etablieren von Richtlinien. Dabei sind gewisse Unternehmen schon weiter als andere, nehmen sich aber trotzdem Zeit, andere bei der Erstellung von Richtlinien zu unterstützen.

Die dritte Möglichkeit der Zusammenarbeit besteht darin, dass Unternehmen oder Behörden aktiv mit anderen in OSS-Projekten zusammenarbeiten. Dabei werden finanzielle und/oder personelle Ressourcen aufgewendet. Wie oben erwähnt, kann man sich durch ein Sponsoring finanziell für eine bestimmte Lösung und Zeit beteiligen. Eine andere Form der Zusammenarbeit ist, dass man einer Community längerfristig beitritt, sich finanziell an Projekten beteiligt oder durch personellen Aufwand in Form von Fehlermeldungen oder Funktionserweiterung etwas an die Community zurücksteuert und sich auf diese Weise an der Weiterentwicklung eines Projekts beteiligt. Zudem gibt es die Möglichkeit, dass ein Unternehmen oder eine Behörde eine Community gründet. Dabei wird vom Unternehmen eine führende Rolle übernommen, indem aktives Community Management betrieben wird und OSS-Projekte realisiert werden. Einerseits stellt man selbst finanzielle und personelle Ressourcen zur Verfügung, andererseits mobilisiert man andere, sich finanziell und personell zu beteiligen. Zum Teil kann es jedoch ein Hindernis bei der Mobilisierung anderer darstellen, wenn die Community im Namen eines Unternehmens oder einer Behörde gegründet wurde. In solchen Fällen gibt es die Möglichkeit, sich mit einer Non-Profit-Open-Source-Organisation wie der *Eclipse Foundation* oder der *Linux Foundation* zusammenzutun, um auf einem neutralen Grund unternehmensübergreifend an OSS-Lösungen zusammenzuarbeiten.

6 Schlussfolgerung

In diesem letzten Abschnitt werden die wichtigsten Ergebnisse zusammengefasst. Des Weiteren werden praktische Implikationen der gefundenen Ergebnisse für Behörden und Unternehmen erläutert, sowie die Limitationen der vorliegenden Arbeit und Möglichkeiten für zukünftige Forschung im Bereich der OSS-Veröffentlichung.

6.1 Zusammenfassung

OSS hat in den letzten 50 Jahren einen Wandel von einer Nischenerscheinung von intrinsisch motivierten Entwicklern zu einem festen Bestandteil der Innovationsstrategie grosser IT-Unternehmen durchlebt (Schrape, 2016, S.9). Die vorliegende Studie zeigt auf, dass der Wandel von OSS weitergeht, weil nun auch Unternehmen oder Behörden in der Schweiz, welche nicht aus der IT-Industrie stammen, OSS nutzen und darüber hinaus anfangen eigene Codes oder Projekte unter einer OSS-Lizenz zu veröffentlichen. Treiber für die Veröffentlichung von OSS durch die neuen Akteure waren in den meisten untersuchten Fällen einzelne Mitarbeitende, weshalb von einer *Bottom-up* Initiative gesprochen werden kann. In einem Fall lag eine *Top-Down* Initiative vor, wo die Politik den Anstoss für die Veröffentlichung gegeben hat.

Die Gründe für die Veröffentlichung wurden in den einzelnen Fallstudien in Kapitel 4 und in 5.1 erläutert. Wie auch in früheren Studien mit IT-Unternehmen, wurde der Nutzen von OSS Veröffentlichungen von den neuen Akteuren in der Schweiz vor allem in folgenden Punkten gesehen: die Unabhängigkeit gegenüber grossen Softwareherstellern, die Rekrutierung von IT-Spezialisten, die Förderung von Schnittstellen und die Verbesserung der Software. Im Gegensatz zu früheren Studien wurde der Verkauf von komplementären Dienstleistungen oder die Steigerung der Innovation durch die Veröffentlichung nicht genannt. Die Kostensenkung wurde in manchen Fällen als langfristiges Ziel genannt, stellte jedoch, im Gegensatz zu früheren Studien, kein Hauptargument dar (Andersen-Gott et al., 2012; Bonaccorsi & Rossi, 2006; Wichmann, 2002).

In der vorliegenden Studie konnten zudem neue Nutzen der Veröffentlichung von OSS identifiziert werden. Insbesondere der Reputationsgewinn durch die Veröffentlichung von OSS war ein Grund, welcher häufig genannt wurde. Der

Reputationsgewinn dient demnach nicht, wie in einer früheren Studie (Bonaccorsi & Rossi, 2006, S.48), ausschliesslich der Rekrutierung neuer Mitarbeitender, sondern der generellen Steigerung der Arbeitgeberattraktivität und der Schaffung von Transparenz. Die Arbeitgeberattraktivität umfasst bei den neuen Akteuren auch die Mitarbeiterentwicklung bestehender Mitarbeiter. Die Schaffung von Transparenz umfasst verschiedene Ziele. Ein Ziel liegt in der Transparenz gegenüber den Bürgern, damit sie sehen wie öffentliche Gelder eingesetzt und wie Daten verarbeitet werden. Anderseits kann die erhöhte Transparenz Unternehmen dabei unterstützen, Akzeptanz in der IT-Industrie zu erlangen, weil die befragten Unternehmen als Versicherungen oder Finanzdienstleister dort nicht den besten Ruf geniessen. Auch fühlen sich die befragten Mitarbeiter der verschiedenen Unternehmen moralisch dazu verpflichtet, einen Beitrag zu OSS zu leisten und nicht nur zu konsumieren. Dies stützt die These von Andersen Andersen-Gott et al. (2012), wonach es eine Verschiebung in der Sichtweise kommerzieller Unternehmen auf OS gibt.

Die vorliegende Arbeit zeigt des Weiteren auf, dass OSS bei keinem der befragten neuen Akteure strategisch offiziell verankert ist. Eine Erklärung für diese fehlende strategische Verankerung ist, wie in früheren Studien bereits beschrieben (Hauge et al., 2010; Ven & Verelst, 2006, S.117), dass es einzelne Mitarbeitende des Unternehmens oder der Behörde waren, die das Veröffentlichen von OSS *Bottom-Up* getrieben haben und es sich bei diesem Prozess nicht um eine strategische Entscheidung des Managements handelte. Ein weiterer Grund für die fehlende Strategie bei den neuen Akteuren liegt darin, dass die Softwareentwicklung nicht zum Kerngeschäft der Befragten gehört. Zudem befinden sich die befragten Unternehmen allesamt erst am Anfang dieses Prozesses. Dieser muss sich zuerst noch etablieren, bevor die Entscheidungsträger involviert werden. Im Fall von KAIO handelte es sich zwar um eine *Top-down* Entscheidung, jedoch ist auch in diesem Fall die strategische Bedeutung klein, weil die Veröffentlichung von OSS auf freiwilliger Basis geschieht und die Dienstleistung vom KAIO eine Holschuld der Ämter und Direktionen des Kantons Bern darstellt.

Wird das vorgestellte Maturitätsmodell, welches Interaktionen von Unternehmen mit OSS erklärt (vgl.2.3.2), auf die untersuchten Fälle transferiert, lässt sich feststellen, dass die befragten Unternehmen und Behörden einen unterschiedlichen Grad an

Maturität aufweisen. Alle haben die Stufe eins «Use» erreicht, weil bei allen OSS genutzt wird.

Das Ziel der Helvetia Versicherung und SIX besteht darin, die nächste Stufe *«Contribute»* zu erreichen, indem sie vom passiven Benutzer zum aktiv Beitragenden übergehen wollen. Damit die Unternehmen direkt veröffentlichen können und dies nicht über private Accounts von einzelnen Mitarbeitenden gemacht wird, fehlen aktuell jedoch Richtlinien dazu, wie Mitarbeitende im Namen des Unternehmens in anderen Projekten beitragen dürfen.

Das KAIO und das SRF kann der zweiten Stufe *«Contribute»* zugeteilt werden, weil beide mindestens ein Projekt veröffentlicht haben, sie aber nicht bei anderen OSS-Communities beitragen.

Die Baloise kann der dritten Stufe *«Champion»* zugeordnet werden, weil sie vom passiven Benutzer zum aktiv Beitragenden übergegangen ist. Sie verwaltet einen *GitHub* Account als Unternehmen, bringt sich in anderen Communities durch Fehlermeldungen ein, veröffentlicht kleinere Hilfsprojekte oder beteiligt sich durch Sponsoring. Ziel von der Baloise ist es die weiteren Stufen zu erreichen, indem ein strategisches Projekt veröffentlicht wird.

Swisstopo war von den befragten neuen Akteuren am weitesten und kann zwischen der dritten Stufe *«Champion»* und der vierten Stufe *«Collaborate»* eingeordnet werden. Swisstopo übernimmt Verantwortung, indem sie eine führende Rolle übernimmt, aktiv Quellcodes veröffentlicht, OSS-Projekte leitet, eine eigene Community aufgebaut hat und in anderen Communities aktiv ist, eigenes Kapital einsetzt und durch ein Crowdfunding, Kapital zur Realisierung von OSS-Projekten sammelt.

Auch die interne Organisation und die Richtlinien für das Veröffentlichen von OSS war bei den befragten Unternehmen und Behörden unterschiedlich.

Bei der Helvetia Versicherung arbeiten einzelne Mitarbeiter neben ihrer offiziellen Tätigkeit an OSS Themen und das Ziel besteht darin, Richtlinien für das Veröffentlichen von OSS-Komponenten aufzubauen.

Die SIX verfolgt ebenfalls das Ziel Richtlinien aufzubauen, jedoch geschieht dies nicht neben der offiziellen Tätigkeit, sondern in Form einer offiziellen OSS-Gilde, in

welcher jeder Mitarbeitende eine Stunde pro Woche als produktive Zeit anrechnen kann.

Die Baloise hat Richtlinien für das Veröffentlichen von OSS erarbeitet und muss nun schauen, ob sich diese bewähren. Zudem können die Mitarbeitenden bei der Baloise einen Tag im Monat an Themen arbeiten, welche sie frei wählen können. So wurden bereits einige kleine OSS-Projekte realisiert. Zwei Mitarbeiter bilden ein inoffizielles OSS-Team, welches das Thema OSS im Unternehmen vorantreibt.

Beim SRF sind die Ressourcen für OSS begrenzt, denn das ganze Team für Datenjournalismus besteht aus nur drei Mitarbeitern, welche bei gewissen Publikationen auch Daten und Codes mitveröffentlichen. Innerhalb des SRF Data Teams wurden inoffizielle OSS Richtlinien erarbeitet.

Beim KAIO wurden OSS Richtlinien erarbeitet. Aktuell bewirtschaftet eine Person das Thema OSS, indem eine Initialberatung angeboten wird, durch welche die Ämter und Direktionen bei der Veröffentlichung von eigener OSS unterstützt werden. Swisstopo hat ein ganzes Team mit ungefähr zehn Mitarbeitenden, welche sich hauptsächlich mit OSS-Projekten beschäftigen. Die expliziten Richtlinien für OSS stehen in der AGB des Bundes. Swisstopo war der einzige Fall, bei welchem Mitarbeiter die Möglichkeit haben, offizielle OSS-Schulung zu besuchen. Bei den anderen Fällen sind OSS-Schulungen geplant.

Während die Einflussnahme in OSS-Communities in den vergangenen 20 Jahren zu einem integralen Bestandteil von Unternehmen in der Softwareindustrie geworden ist (Schrape, 2016, S.72) hält sich die Einflussnahme in OSS- Communities bei den befragten Unternehmen und Behörden noch in Grenzen. Eine Erklärung dafür ist, dass bei den meisten befragten Unternehmen und Behörden die Veröffentlichung eine neue Erscheinung ist. Aus diesem Grund müssen vorerst intern Richtlinien und Strategien erstellt werden, bevor aktiv mit externen Communities zusammengearbeitet werden kann. Einzig swisstopo betreibt ein aktives Community Management und greift auf kreative Ideen von externen Personen zurück (Dahlander & Magnusson, 2008, S644ff.). Die Entscheidungsfindung innerhalb der selbst aufgebauten Community findet eher zentral statt, trotzdem versucht swisstopo eine Balance zwischen Offenheit und Kontrolle zu finden, weil es wichtig ist die Community zusammenzuhalten.

Neben der aktiven Einflussnahme in einer OSS Community konnten auch andere Formen der Zusammenarbeit beobachtet werden. Dabei stehen die SIX und die Helvetia Versicherung mit anderen Unternehmen im Austausch für die Erarbeitung von Richtlinien. Des Weiteren haben sich die Baloise und die Helvetia Versicherung durch ein Sponsoring an einer OSS-Lösung beteiligt, ohne personelle Ressourcen aufzuwenden. Die Baloise hat zudem temporär für sechs Wochen mit anderen Unternehmen eine OSS-Lösung *Open Prevo* entwickelt, ohne diese zu veröffentlichen. Eine Vision der Baloise besteht in *Open Insurance*, wo mit anderen Versicherungen zusammen unternehmensneutrale OSS-Projekte realisiert werden sollen. Dabei wird die Idee verfolgt, sich mit einer Non-Profit-Open-Source-Organisation wie zum Beispiel der *Eclipse Foundation* oder *Linux Foundation* zusammenzutun, um auf einem neutralen Grund unternehmensübergreifend zusammenzuarbeiten.

6.2 Implikation für Unternehmen und Behörden

Die vorliegende Arbeit soll vor allem Unternehmen, welche nicht mit IT ihr Geld verdienen, aber aktuell OSS passiv nutzen, einen Einblick geben, warum es auch für sie von Vorteil sein kann, aktiv eigene Codes oder Projekte unter einer OSS-Lizenz zu veröffentlichen und welche Risiken dabei entstehen könnten. Die Ergebnisse zeigen, dass es nicht nur einen richtigen Weg gibt, OSS zu veröffentlichen, sondern dass jedes Unternehmen die Auswirkungen der Veröffentlichung von OSS in seinem spezifischen Kontext bewerten sollte. Zudem zeigen die Ergebnisse, dass es verschiedene Arten der Veröffentlichung von OSS gibt und dass bereits kleine Veröffentlichungen eine grosse Wirkung haben können (vgl.2.4.4). Es scheint demnach empfohlen, nicht gleich ein strategisches Projekt zu veröffentlichen, sondern vorerst durch kleine Veröffentlichungen Erfahrungen zu sammeln und als Organisation zu lernen.

Die Ergebnisse zeigen auch, dass das Vorhaben, OSS zu veröffentlichen, oftmals *Bottom-Up* durch Mitarbeitende getrieben ist und keine Management Entscheidung ist, weshalb eine strategische Verankerung in den jeweiligen Unternehmen fehlt. Deshalb ist den Mitarbeitenden von Unternehmen, welche planen OSS zu veröffentlichen, zu empfehlen, alle Stakeholder, wie das Management oder die Rechtsabteilung, frühzeitig einzubeziehen. Dies ist hilfreich, um die Rahmenbedingungen und die Einflussfaktoren – wie den Markt, die Konkurrenten oder die Kundenstruktur - zu

analysieren und dementsprechend eine geeignete OSS-Strategie aufzustellen. Um zusätzliche Unterstützung zu erhalten, sollten auch die Personal- und Marketing-Abteilungen eingebunden werden, denn durch die Veröffentlichung von eigenen Projekten kann gemäss den Ergebnissen ein Reputationsgewinn für das Unternehmen erzielt werden. Durch eine zielgerichtete Kommunikation dieser Abteilungen kann der Ruf des Unternehmens und die Arbeitgeberattraktivität somit gesteigert werden.

Die Mitarbeitenden der untersuchten Unternehmen geniessen unterschiedlich viel Freiheiten für die Arbeit an OSS-Themen. Damit die intrinsische Motivation der Mitarbeitenden gefördert werden kann, ist dem Management zu empfehlen, seinen Mitarbeitenden genügend Freiheiten zu gewähren, damit Grenzen überbrückt werden können und sie das Unternehmen mit neuen Innovationen, wie der Veröffentlichung von OSS, in Kontakt bringen können.

Auf der anderen Seite zeigt die vorliegende Studie auf, dass es auch für Behörden sinnvoll ist, eigene Software unter einer OS-Lizenz zu veröffentlichen (vgl. 2.5.1) und dies von verschiedenen Behörden in der Schweiz auch schon erfolgreich praktiziert wird. Das Beispiel des Kantons Bern zeigt, dass durch politisches Lobbying in Kombination mit Entwicklung einer OSS-Dienstleistung durch das KAIO, die Rahmenbedingungen geschaffen und eine erste Applikation veröffentlicht wurde. Nachdem diese Rahmenbedingungen im Kanton Bern geschaffen wurden, besteht die Herausforderung darin, diese Möglichkeit den Ämtern und Direktionen des Kantons bekannt zu machen. Vor allem bei Neuentwicklungen sollte eine OSS-Lösung in Kombination mit einer Veröffentlichung in Betracht gezogen werden. Aktuell beschränkt sich der Service nur auf die kantonale Ebene, jedoch könnten auch die Gemeinden im Kanton Bern davon profitieren, weil viele von ihnen ähnliche Bedürfnisse haben und dementsprechend ähnliche Software im Einsatz ist.

Während im Kanton Bern eine Rechtsgrundlage geschaffen wurde, zeigt die Analyse dieser Studie auch auf, dass auf Bundesebene für die Veröffentlichung von OSS eine klare Rechtsgrundlage fehlt und ein Entscheid des Bundesrates aktuell noch ausstehend ist (vgl.2.5.2). Damit weitere Bundesbehörden dem Beispiel von swisstopo folgen können, ist es wichtig, dass sie eine Rechtsgrundlage dafür haben. Es scheint zudem sinnvoll, dass die Veröffentlichung von OSS auf Bundesebene geregelt würde und nicht jeder Kanton einzelne Regelungen treffen müsste, denn vom politischen Vorstoss bis zur Umsetzung sind im Kanton Bern mehr als fünf Jahre vergangen. Die

vorliegenden Ergebnisse lassen vermuten, dass es sich für alle Bundesstellen, Kantone und Gemeinden in der Schweiz empfiehlt, den Weg der Veröffentlichung eigener Software zu gehen. Die Grundlage in Form von Checklisten, Richtlinien und Empfehlungen wurde vom KAIO auf *GitHub* publiziert. Damit müssten sie einzelnen Behörden das Rad nicht mehr neu erfinden und könnten nach dem OSS-Gedanken Wissen miteinander teilen. Je mehr Akteure diesen Weg gehen, desto grösser könnte die Zusammenarbeit über die Behörden hinweg werden, was in einem Nutzen für alle Beteiligten resultieren würde.

6.3 Limitation und zukünftige Forschung

Die Ergebnisse der vorliegenden Arbeit sollen mittels eines explorativen Fallstudienansatzes erste Erklärungen eines zeitgenössischen Phänomens in einem realen Umfeld liefern. Warum neue Akteure in der Schweiz OSS veröffentlichen, wurde bislang kaum erforscht. In dieser Arbeit wurden sechs Unternehmen und Behörden aus der Schweiz befragt. Die untersuchten Unternehmen und Behörden stellen jedoch nur eine kleine Teilmenge möglicher Fallstudien dar und zeigen daher nicht alle möglichen Gründe und Vorgehen neuer Akteure bei der Veröffentlichung von OSS auf. Aufgrund eines Mangels an Zeit und Ressourcen, konnten in dieser Arbeit nicht mehrere Datenerhebungsmethoden eingesetzt werden, auch wenn dies erkenntnistechnisch wertvoll gewesen wäre. Es wurden ausschliesslich qualitative Daten in Form von Experteninterviews analysiert. Die durchgeführten Experteninterviews widerspiegeln hauptsächlich die Entwicklerperspektive, weshalb die Ergebnisse mit Vorsicht zu interpretieren sind.

Trotz der genannten Limitationen zeigt diese Arbeit auf, dass es einen Trend zur Veröffentlichung von OSS in verschiedenen Branchen und Bereichen in der Schweiz gibt.

Für die zukünftige Forschung wäre es interessant, diesen Trend auf internationaler Ebene zu vergleichen. Insbesondere wäre spannend zu untersuchen, wie die Rechtsgrundlagen für das Veröffentlichen von OSS in anderen Ländern für die Behörden aussehen und wie diese geschaffen wurden. Des Weiteren scheint es von praktischer Relevanz zu untersuchen, welche Rechtsgrundlagen geschaffen werden müssen, damit die Behörden international zusammenarbeiten können. Zudem wäre es spannend die Managementperspektive bezüglich der Veröffentlichung von OSS in

Unternehmen ausserhalb der IT-Industrie genauer zu untersuchen. Insbesondere wäre es dabei interessant, den wahrgenommenen Nutzen und die Risiken durch das Management mit denjenigen durch die Entwickler wahrgenommenen zu vergleichen.

Anhang A

Interview mit Timo Grossenbacher am 17.01.2019

I: Das ganze Interview wird dann als Transkript im Anhang sein und gewisse Teile und Zitate werden in der Arbeit erscheinen.

B: Ja das ist gut, machen wir so.

I: Für unsere Studie haben wir Unternehmen oder Behörden gesucht, welche nicht aus der IT oder Softwareentwicklung stammen, aber aktuell Codes, Daten oder Projekte auf GitHub oder sonst irgendwo mit einer Open Source Lizenz veröffentlichen. So sind wir auf SRF gestossen, weil wir gesehen haben, dass ihr auf GitHub schon einzelne Sachen veröffentlicht habt. Zum Anfang paar einleitende Worte. Mein Name ist Vidhushan Asokan. Ich studiere an der Universität Bern und bin gerade an meinem Masterabschluss. Die Forschungsarbeit ist in Form meiner Masterarbeit. Mein Betreuer ist Matthias Stürmer.

B: Ja, ihn kenne ich.

I: Er beschäftigt sich schon lange mit Open Source Software in der Forschung.

B: Ja genau.

I: Jetzt haben wir neue Akteure entdeckt, welche da mitmachen, die ein anderes Kerngeschäft als IT haben. Aber zuerst mal zu ihnen. Was ist ihre Ausbildung, wie sind SIe zu der SRF gestossen und was ist ihre aktuelle Funktion bei SRF?

B: Ich habe grundsätzlich nicht Journalismus studiert, sondern Geografie. Im Nebenfach hatte ich Informatik. Das ist jetzt ungefähr fünf Jahre her. Dann habe ich kurz beim Tagesanzeiger gearbeitet. Im Jahr 2014 hat man das SRF Data Team gegründet mit drei Leuten. Und wurde angefragt, ob ich da mitarbeiten will. Und bin dann im Oktober 2014 dazugestossen, als das Team gegründet wurde. Das Team gibt es immer noch, bald fünf Jahre und sind immer noch drei Leute. Und wir sind Datenjournalisten. Meine Aufgabe ist mit technischen Mitteln die Automatisierung, Programmierung oder Automatisierung durch Programmierung die Daten journalistisch auszuwerten. Dazu gehört auch Visualisierung, welches ich mittlerweile ein bisschen weniger mache. Das macht mein Kolleg. Mein Kerngebiet sind vor allem die investigativen Recherchen. Nicht unbedingt immer mit Daten, aber oft mit Daten. Und wenn ich mit Daten arbeite, dann arbeite ich mit Scriptsprache und werte die Daten automatisiert aus.

I: Welche Scriptsprachen nutzen sie? Auf GitHub habe ich vor allem R gesehen.

B: Genau es ist eigentlich ausschliesslich R. Also privat und für paar Sachen brauche ich auch Python oder auch Javascript, um zum Beispiel Scraper zu schreiben. Dies geschieht nicht im beruflichen Kontext, aber das wird kommen. Das trifft auch auf die anderen Teammitglieder zu.

I: Wie ist die Idee entstanden, eigene Datensätze zu veröffentlichen und der Allgemeinheit kostenlos zur Verfügung zu stellen?

B: Die Idee ist relativ früh entstanden. Als ich da anfing, habe ich mit R gearbeitet. Irgendwann wurde mir auch bewusst, dass wenn man das alles coded, kann man es auch sehr gut nachvollziehen. Ich habe mir damals gar nicht so gross Gedanken darüber gemacht, wieso man das eigentlich machen sollte. Im Sommer 2016 vielleicht auch schon ein bisschen früher, haben wir effektiv angefangen die Meinung zu bilden, dass man es eigentlich auch veröffentlichen könnte. Alle diese Daten, welche grundsätzlich nicht irgendwie den Datenschutz tangieren und man veröffentlichen dürfte. Und all diese Methoden, welche wir selbst schreiben, wie diese Scripts, wieso sollte man diese nicht veröffentlichen. Und ohne gross zu überlegen, sage ich jetzt mal. Wir haben das abgeklärt mit unserem Rechtsdient. Wir haben dann auch einen Haftungsausschluss, vielleicht kommen wir

noch dazu über den zu sprechen, der war eigentlich schon von Anfang ein drin. Und dann haben wir das so veröffentlicht, ja. Aber jetzt eine riesen Aufzählung von Gründen, haben wir uns eigentlich im Nachhinein überlegt. (...). Wobei ich muss schon noch sagen, da steht so ein Text auf unserer GitHub Seite "Legt Wert darauf, dass die Daten vor Prozessierung und Analyse nachvollzogen und überprüft werden kann". Zum anderen soll es Dritten ermöglicht werden auf diese Vorarbeit aufzubauen. Also das ist eigentlich ein Text, welchen wir im Jahr 2016 hingeschrieben haben. Das mussten wir uns also schon damals überlegt haben. Die Kerngründe waren damals schon vorhanden.

I: Ihr habt jetzt alles auf GitHub veröffentlicht oder habt ihr noch andere?

B: Nein ist eigentlich alles ausschliesslich auf GitHub. Kommt in ganz wenigen Fällen vor, dass wir die Daten sammeln und aus irgendwelchem Grund gibt es kein R Script. Weil vielleicht Personen, welche daran gearbeitet haben, zum Beispiel Praktikanten oder irgendwelche Redaktionsgäste, die vielleicht nicht mit R programmieren können. Und dann gab es gar kein Script oder weil vielleicht auch die Datenauswertung trivial war. Dann kann es auch vorkommen, dass wir zum Beispiel die Daten in ein Google Drive Spreadsheet stellen und dann so zum direkten Download anbieten, aber das ist sehr selten der Fall.

I: Und was für Projekte oder Datensätze habt ihr bereits veröffentlicht?

B: Ja, das sieht man da auf dieser Übersichtsseite. Das sind ein bisschen mehr als ein Dutzend Projekte. Es fängt an mit Wahlprojekte 2015, die haben wir im Nachhinein veröffentlicht. Und 2016 haben wir sie alle veröffentlicht. Ja, verschiedene Recherchen, da ist inhaltlich Kraut und Rüben gemischt. Zum Beispiel Kriegsmaterialexport, da haben wir viel Daten vom Staatssekretariat für Wirtschaft auswertbar gemacht, ausgewertet und zur Verfügung gestellt. Die Daten gab es nur als PDFs und wir haben sie dann maschinenlesbar gemacht. Ein Kollege hat etwas über Roger Federer gemacht und er hat da extrem viel Tennisstatistiken zusammengetragen, welche für sich eigentlich auch schon unter einer öffentlichen Lizenz veröffentlicht wurden, aber die Aufbereitung relativ kompliziert waren. Die ganze Aufbereitung und die Daten selbst hat er dann auch wiederum veröffentlicht. Also es ist wirklich inhaltlich Kraut und Rüben.

I: Wie entscheidet ihr, welche Datensätze oder Projekte veröffentlicht werden?

B: Also die Frage ist eher, wie entscheiden wir, dass die Daten nicht veröffentlicht werden. Bei uns ist schon der Regelfall, dass Daten veröffentlicht werden. Wenn gewichtige Gründe dagegensprechen, dann machen wir es nicht. Der wichtigste Grund ist sicher, wenn es in den Daten persönlich identifizierbare Records hat, also persönlich identifizierbare Daten. Denn es wäre gegen das Datenschutzgesetzt einerseits. Ausser es handelt sich vielleicht um öffentlich bekannte Personen, wo es ein öffentliches Interesse besteht, dass die Daten öffentlich sind. Aber auch dann sind wir sehr zurückhaltend. Das ist eigentlich der Hauptgrund. (...). Ja vielleicht ist noch ein Grund, wenn eine Auswertung so trivial ist, dann hat da niemand Interesse daran, wenn jetzt der Code öffentlich ist. Datenschutz ist eigentlich der einzige Grund etwas nicht zu veröffentlichen. Oder wenn wir jetzt ein Abkommen haben, dass wir Daten von jemandem exklusiv erhalten. Fragen wir sie an, ob es in Ordnung ist, wenn wir die Daten so weiter veröffentlichen oder zumindest das was wir daraus gemacht haben. Und wenn sie sagen „nein ist für uns nicht in Ordnung“, dann machen wir es auch nicht. Aber wir versuchen sie dann aber auch davon zu überzeugen, dass es schlussendlich Allen dient. Aber manchmal funktioniert es halt einfach nicht. Das ist auch ein Ausnahmefall.

I: Wer entscheidet schlussendlich?

B: Wir im Team.

I: Vom SRF Data Team?

B: Genau. Es ist meistens eine Teamdiskussion. Wenn es jetzt wirklich umstritten wäre, vor allem aus rechtlicher Sicht, dann würde man wahrscheinlich noch den Rechtsdienst vom SRF konsultieren. Aber das war bis jetzt noch nie der Fall.

I: Wann ist der beste Moment für die Veröffentlichung?

B: Ja ich denke in der Regel streben wir an, die Daten direkt mit der journalistischen Publikation mitzuveröffentlichen. Dann können die Leute, wenn sie auf Srf.ch gehen, die Artikel lesen oder wenn sie etwas im 10 vor 10 oder wo auch immer sehen, die Daten von

Anfang an finden. Nicht vorher, aber gleichzeitig mit der Publikation. Was manchmal auch vor kommt, weil man keine Zeit hat, dass es auch ein paar Tage später publiziert wird. Aber das ist auch die Ausnahme

I: In der Open Source Welt ist es ja auch gang und gäbe, dass man es öffentlich entwickelt und es der Öffentlichkeit zur Verfügung stellt. Bei ihnen ist es eher so, dass ihr Inhouse entwickelt und erst danach veröffentlicht?

B: Genau. Das hat einfach typische journalistische Gründen. Der Journalismus öffnet sich zwar auch immer mehr, es gibt auch immer mehr Crowdsourcing Projekte, in welchen man vielleicht schon über den Stand der Recherche informiert. Aber da wir uns oft auch im investigativen Bereich bewegen, wo es vielleicht darum geht eine neue Erkenntnis über etwas zu erkennen. Und vielleicht auch exklusiv neue Informationen zu erhalten, ist es auch unser Interesse, dass nicht jeder weiss, an was wir gerade dran sind. Ich denke es ist ein legitimes journalistisches Interesse und das ist auch oft der Grund wieso wir nicht irgendetwas, sage jetzt mal, unfertiges veröffentlichen. Was man auch noch sagen muss: Ich denke es geht vielen Softwareentwickler so oder Leute die grundsätzlich programmieren. Wir versuchen schon von Anfang an quasi elegant und lesbar zu programmieren. Oft probiert man auch einfach aus und dann legt man keinen Wert auf die Lesbarkeit und Nachvollziehbarkeit. Darum würde es sich nicht eignen, von Anfang an einen Einblick zu gewähren. Das würde auch nichts bringen.

I: Kam die Idee der Veröffentlichung vom SRF Data Team oder ist es vielleicht aus dem Management gekommen?

B: Nein das ist wirklich von uns intrinsisch, also eigentlich von mir ist die Idee gekommen, dass wir das machen könnten.

I: Wieso habt ihr GitHub ausgewählt und nicht die SRF Homepage?

B: Das Ding ist natürlich, dass wir schon vor diesen Veröffentlichungen, schon im Jahr 2014 und 2015, von Anfang an mit GitHub gearbeitet haben, weil wir meistens mehrere Leute an einem Projekt arbeiten. Und das waren halt die klassischen Gründe wieso man GitHub verwendet. Klar, man könnte auch etwas Anderes verwenden, wie Bitbucket oder GitLab oder was auch immer. Irgendwie haben wir uns mal auf GitHub entschieden. Da wir sowieso unsere Sachen auf dieser Plattform hosten, war es danach naheliegend, dass wir es auch dort veröffentlichen.

I: Musstet ihr das Management überzeugen, damit ihr es veröffentlichen durftet und wie haben sie auf dieses transparente Vorgehen reagiert?

B: Grundsätzlich sind wir zum Rechtsdienst gegangen, weil wir es rechtlich abklären wollten und der Rechtsdienst hat da keine Einwände gesehen. Ausser, dass man einen Haftungsausschluss beifügt und ja das war eigentlich der Prozess. Das Management hat das natürlich mitbekommen. Ich kann mich ehrlich gesagt nicht mehr daran erinnern, ob man das Management auch zu dem Zeitpunkt konsultiert hat. Aber schlussendlich hat es da nie einen Widerspruch gegeben zu dieser Praxis. Es kann gut sein, dass wir zum Management sind und sie gefragt haben, aber das weiss ich ehrlich gesagt nicht mehr. Ich weiss einfach nur noch das mit dem Haftungsausschluss.

I: Also ihr hattet in dem Fall mit der Idee keine Schwierigkeiten gehabt?

B: Nein eigentlich nie.

I: Ansonsten wüsstet ihr es wahrscheinlich noch?

B: Ja, auf jeden Fall. Aber wir haben auch von Anfang an auch mit dem Rechtsdienst gearbeitet. Und wir haben auch gesagt, dass es nicht das Ziel ist, irgendwelche sensitiven Informationen zu veröffentlichen.

I: Was ist die Motivation von SRF, die eigenen Datensätze zu veröffentlichen?

B: Es gibt eigentlich zwei Hauptmotivationen. Erstens finden wir, wenn man sozusagen Wissenschaft betreibt, schafft man ja Wissen ein Stückweit. Man arbeitet vielleicht nicht immer hundertprozentig nach den gleichen wissenschaftlichen Kriterien. Aber auf eine Art ist es gleich wissenschaftlich. Dann sind wir der Meinung, dass es nachvollzierbar und transparent sein muss. Vor allem wenn man Codes schreibt, trifft man eigentlich auch viele Annahmen. Also wenn man Daten analysiert, grundsätzlich trifft man Annahmen und man trifft Entscheidungen. Und wir finden, dass diese Entscheidungen transparent sein müssen, dass man das hinterfragen und kritisieren kann.

Sonst kann man es ja gar nicht hinterfragen, was wir machen. Das ist mal der eine Grund. Der zweite Grund ist (...) ein gesellschaftlicher Grund. SRF ist Teil des Service Public ist und wir es auch eine Art Service Public ansehen, dass das was wir produzieren, auch anderen zu Gute kommen könnte.

I: Welche strategische Bedeutung hat jetzt Open Source Software oder die Veröffentlichung für SRF? Ist es irgendwo in einer Strategie oder IT Strategie implementiert?

B: Nein, das glaube ich weniger. Da bin ich ehrlich gesagt ein wenig überfragt. Es ist jetzt nicht Teil von irgendwelcher offiziellen Strategie des Managements oder so. Weil das Volumen, welches wir veröffentlichen, ist ja nicht riesig. Zumindest in der Szene, also Szene, das heisst Open Data Community und andere Journalisten, ist es bekannt, dass wir das machen und wird auch geschätzt. Und es ist sicher auch ein gutes Aushängeschild. Aber offiziell glaube ich nicht, dass es Teil einer Strategie ist. Da müsste man wahrscheinlich jemanden anderes fragen.

I: Kommen wir zu der internen Organisation. Ihr habt vorher erwähnt, dass ihr drei Leute seid im SRF Data Team. Haben noch weitere Leute Zugriff auf den GitHub Account?

B: Ja, sicher die drei Kernmitarbeiter sage ich jetzt mal. Und natürlich auch wenn Praktikanten oder Redaktionsgäste dabei sind. Zum Teil arbeiten wir auch mit externen Firmen, vor allem im Bereich der Datenvisualisierung. Die haben dann temporär auch Zugriff auf gewisse Repositories. Wir sprechen jetzt von den privaten Code Repositories, oder?

I: mhm (bajahend)

B: Ja sicher immer wir drei und dann temporär noch andere Leute. Jetzt nicht ein riesen Kreis.

I: Könnte man sagen, die drei sind das Kernteam und projektspezifisch können weitere Leute dazukommen.

B: Die verschwinden danach auch wieder. Diese Plätze auf GitHub kosten auch etwas und man kann nicht unbegrenzt Leute hinzufügen. Sonst würde unsere Kreditkarterechnung sehr gross sein.

I: Und es ist so flexibel, dass man jemanden für zwei Monate einen Zugang gewähren kann?

B: Ja auf jeden Fall. Das kann man recht flexibel gestalten bei GitHub.

I: Ich habe eben nur einen privaten Account und keinen Unternehmens Account. Ich habe es für die Studie erstellt, um die Leute zu suchen, aber sonst nicht so viel Erfahrung damit.

B: Es ist relativ simpel, es zu bedienen. Also wenn man eine Organisation hat und nicht einen privaten Account.

I: Habt ihr auch einen privaten Account und was macht ihr dort?

B: Also den privaten Account brauche ich so oder so, sonst könnte ich nicht Teil sein von der Organisation. Also SRF Data ist nicht ein Account im normalen Sinn, sondern ist eine Organisation und dort kommen danach andere Accounts dazu. Also brauche ich auch eigentlich einen privaten. Privat habe ich selbst ein Projekt gemacht. Wir schaffen mit R bzw. R Markdown, um diese Reports zu schreiben. Und ich habe selbst ein Projekt gemacht, mit welchen wir es mit relativ wenig Aufwand veröffentlichen können. Mit dem Projekt haben wir die ganze Veröffentlichung auf GitHub automatisiert. Das ist so ein Skript, welches man laufen lassen kann und danach werden die Sachen automatisch auf eine GitHub Page hochgeladen. Und ich habe auch relativ viel Zeit investiert, um dieses Skript wirklich reproduzierbar zu machen. Man hat ja solche Packages im R gleich wie bei Python und das Problem ist ein bisschen, dass diese Packages laufend weiterentwickelt werden. Und wenn man Packages verwendet vom Jahr 2017, kann es gut sein, dass im Jahr 2019 das Skript zwar noch funktioniert, aber ein leicht anderes Resultat produziert, da die Packages auch leicht anders sind. Es ist auch schon passiert, dass irgendwelche Werte anders berechnet wurden. Das ist der Grund wieso ich danach überlegt habe, wie kann man eigentlich quasi das reproduzierbarer machen, indem man eben diese Softwareversionen wie mitversionieren und danach auch sicherstellen kann, dass man die gleichen Packages verwendet wie früher. Das ist so ein kleines Projekt von mir, welches ich auch privat veröffentliche, quasi metamässig. Es ist nur ein Template, welches wir verwenden und das ist wieder Open Source. Das ist zum Beispiel auf meinem privaten

Account. Ich brauche meinen privaten Account auch für andere Projekten und so.

I: Brauchen sie dafür eine Genehmigung, wenn sie es während der Arbeitszeit machen und veröffentlichen?

B: Nein eigentlich nicht. Also zum Teil entwickle ich daran in meiner privaten Zeit und zum Teil arbeite ich während meiner Arbeitszeit daran, wenn es natürlich etwas ist, was wir bei SRF Data sowieso brauchen. Ob ich es jetzt privat veröffentliche oder nicht, spielt eigentlich nicht so eine Rolle. Es macht einfach Sinn, das als Open Source zu veröffentlichen. Vielleicht wiederrum aus Reputationsgründen, dass man sieht, dass die Arbeit, die wir machen, auch anderen zugutekommt.

I: Ist es geplant, dass euer Team ausgebaut wird? Daten werden ja auch immer wichtiger.

B: Da gibt es aktuell keine Pläne dazu.

I: Kommen wir zur Compliance und Richtlinien. Habt ihr eine offizielle Open Source Compliance? Wissen die Mitarbeiter, was sie dürfen und was nicht? Gibt es dafür ein Regelwerk?

B: Ja. Teamintern haben wir informelle Guidelines für unser Template oder für Sachen, was wir unter SRF Data veröffentlichen. Also diese sind aufgeschrieben und dort stehen verschiedene Sachen drin. Wie zum Beispiel, dass man den Code kommentieren soll, keine sensitiven Informationen wie Passwörter im Code haben und so weiter. Und was wir sicher auch haben, bevor wir Sachen veröffentlichen, gibt es immer das Vier-Augen-Prinzip. Also wir versuchen diesen Code nochmals auf einer anderen Architektur laufen zu lassen. Also vielleicht wird er auf Mac programmiert, jemand probiert es dann noch auf Linux oder Windows aus und liest es durch. Da haben wir eigentlich intern schon eine Compliance auf jeden Fall, aber nicht hoch formell. Ich bin mir jetzt nicht sicher, ob andere Stellen bei SRF Codes auch noch Open Source veröffentlichen. Das kann sehr gut sein und was die für Compliance haben, das wüsste ich jetzt nicht.

I: Diese Richtlinien gelten also für euer Team und nicht für die gesamte SRF Gruppe?

B: Nicht das ich wüsste.

I: Und welches Team könnte sonst noch Open Source veröffentlichen?

B: Ja, es gibt bei SRF die Abteilung SRF Online. SRF Online sind eigentlich zuständig für Content Management System, für die verschiedene Apps etc. Und es kann sehr gut sein, dass die auch Sachen Open Source veröffentlichen.

I: Haben die einen GitHub Account?

B: Ja, die haben glaube ich einen GitHub Account. (sucht im Internet). Weiss gerade nicht wie dieser Account heisst.

I: Ich habe nur euren Account gefunden.

B: Ja, das ist auch eine Organisation. Die Organisation heisst SRF Online und die URL ist mmz-srf. Ich habe hier Zugriff und bin Member dieser Organisation, aber was sie Open Source veröffentlichen, weiss ich nicht.

I: Ist die Zusammenarbeit in dem Fall sehr gering?

B: Ja, die ist sehr gering. Also ja eher situativ, aber dann eher im Frontend Bereich. Wenn es darum geht zum Beispiel Visualisierungen zu programmieren und irgendetwas im Zusammenspiel mit dem CMS nicht funktioniert, dann gibt es sicher Austausch.

I: Aber ihr habt jetzt nicht eine gemeinsame Compliance, welche von der Dachorganisation wäre?

B: Nein.

I: Und ist somit auch nicht strategisch eingebettet?

B: Nein.

I: Reden wir nochmals über die Prozesse. Bei Entwicklung haben Sie ja vorher erklärt, dass dies nicht öffentlich stattfindet. Können sie erklären wie das SRF Data Team schrittweise bei der Veröffentlichung und bei der Wartung vorgeht?

B: Also grundsätzlich arbeiten wir mit GitHub. Es arbeiten verschiedene Leute daran. Dann gibt es irgendeinmal einen Veröffentlichungstermin. Dann wird entschieden ob wir den Code und die Daten mitveröffentlichen, nur den Code oder nur die Daten veröffentlichen. Aber in der Regel veröffentlichen wir Code und Daten. Danach

machen wir eine Kopie vom privaten Repository, gehen durch unsere Guidelines durch, entfernen alles, wo wir das Gefühl haben, dass es nicht wichtig ist oder nichts mit der Kernrecherche zu tun, welche wir dann veröffentlichen. Dann gibt es einen Peer Review Prozess, wenn es ihn nicht vorher schon sowieso schon gegeben hat. Dann wird es nochmals zusammen angeschaut. Danach haben wir wie erwähnt ein Deploymentscript, welche dann quasi das R Mardown in ein HTML umwandelt und das HTML automatisch auf GitHub hochlädt. Das ist eigentlich dieser Deployment Prozess. Das ist eigentlich relativ simpel. Danach ist einerseits der Code in einem Public GitHub Repository inklusiv die Daten. Das Markdown als HTML File ist auf so einer GitHub Page publiziert. Das ist für Leute, welche keine Lust haben, das GitHub Repository herunterzuladen und es selber zu machen. Und als dritte Komponente wird auch noch ein gezippter Ordner geschaffen, wo man quasi den Code und Daten einfach so herunterladen kann, falls man mit GitHub überhaupt nicht zu Schlage kommt. Also für die, welche GitHub überhaupt nicht kennen, nicht wissen wie man etwas klonet oder so. Für die haben wir dann noch diesen Zip Ordner

I: Wie sie vorher erwähnt haben, beziehen sich die Guidelines auf die sensitiven Daten, oder?

B: Ja, das ist drin. Aber das wird auch schon bei dem Entscheid, überhaupt Daten zu veröffentlichen, überlegt. Mit sensitiven Daten in den Guidelines meine ich eher, dass zum Beispiel keine Passwörter für irgendwelche Accounts oder APIs drin sind.

I: Erhält ihr auch externe Anfragen über eure Datenpakete?

B: Ja auf jeden Fall. Es kommt ab und zu vor, nach einer Recherche, dass es heisst, wir wären interessiert an diesen Daten und so. Dann kommt zum Teil auch die Frage, ob man die abkaufen könnte. Wenn wir diese sowieso schon veröffentlicht haben, dann sagen wir das natürlich und verweisen die Leute darauf. Wenn wir es nicht veröffentlicht haben, dann geben wir es auch nicht weiter. Entweder veröffentlichen wir es auf GitHub für alle oder gar nicht.

I: Aber erhält ihr auch spezifische Fragen, wie zum Beispiel man die Pakete anwenden kann? Habt ihr solche anfragen?

B: Ja, so Supportanfragen, dass es irgendwelche Probleme gibt?

I: Genau.

B: Ja, das ist relativ selten der Fall. Dann kann man ein stückweit Hilfe leisten, aber wir haben auch die Devise, dass wir dann nicht wochenlang Support leisten, da dies nicht wirklich drin liegt. Wir versuchen halt den Code so verständlich wie möglich machen und sagen auch ganz klar, wenn man etwas machen will mit dem Code, dann muss man auch gewisse Voraussetzungen haben. Dann muss man R kennen und so. Wir können den Leuten dann nicht noch das R beibringen oder so, das geht aus Ressourcengründen nicht.

I: Habt ihr einen Prozess, wie ihr mit Anfragen umgeht oder ist das individuell?

B: Das ist individuell.

I: Seht ihr noch einen Bedarf die Veröffentlichung zu optimieren?

B: Nein, da wir eigentlich im Moment die Scripts wirklich reproduzierbar machen können, dass sie auch noch in paar Jahren funktionieren, da fand eigentlich kontinuierlich immer wieder eine Optimierung statt. Es ist eigentlich ein Ongoing Projekt. Also ich würde sagen, jetzt im Moment sind wir mit dem aktuellen Stand sehr zufrieden. Und wir können auch nicht unbegrenzt Ressourcen in das investieren, es muss sich auch irgendwodurch amortisieren, sage ich jetzt mal.

I: Und wie würdet ihr dies amortisieren?

B: Amortisieren im Sinne, wenn man den Gesamtaufwand betrachtet, welcher in die Veröffentlichung geht, sprich die Entwicklung dieser Templates, das sind vielleicht 100 Stunden oder so. Je mehr Projekte wir mit diesen Templates veröffentlichen können, desto besser ist es amortisiert, sage ich jetzt mal. Also nicht, dass wir irgendwie 100 Stunden in ein Template investieren, welches wir dann nur einmal verwenden. Das ist so ein bisschen die Überlegung.

I: Brauchen sie auch externe Open Source Software Komponente?

B: Ja, auf jenen Fall. Eigentlich alles. Wir brauchen eigentlich nur externe Open Source Komponenten. Bei R ist sowieso alles Open Source. Alle diese Packages sind Open Source. Danach in der Frontendentwicklung ist

eigentlich auch alles Open Source. Proprietäre Software verwenden wir eigentlich fast nie.

I: Haben sie Richtlinien, wie die externen Komponenten bei euch integriert werden?

B: Bei uns jetzt nicht. Also klar, dann gibt es wieder informelle Richtlinien. Dass man beim R zum Beispiel Packages nimmt, wo man weiss, dass sie gut, tried and testet sind. Da gibt es eine gewisse Community und eine Dokumentation dazu. Ausser wenn wir wirklich kein besseres Package findet, kann man etwas verwenden, das nicht so gut dokumentiert ist. Aber da ist schon die Idee, dass man auch möglichst wenig externe Packages braucht, sondern versucht mit denen, die man hat, möglichst viel zu machen. Das betrifft die Analyse mit R und eben die Frontendentwicklung. Aber eben, es gibt keine formellen Guidelines.

I: Verwendet ihr Code Scanning Tools? Wie stellt ihr sicher, dass die Packages unter einer Open Source Lizenz steht?

B: Grundsätzlich finde ich, ist es gar nicht nötig, weil was auf Cran veröffentlich wird, also auf dem Archiv von R. Da muss ich ihnen ehrlich sagen, ich weiss gar nicht, wie dies funktioniert mit den Lizenzen. Ich bin jetzt davon ausgegangen, was dort und auf NPM veröffentlicht wird, genutzt werden kann.

I: Was ist das?

B: NPM ist Node Package Manager. Den braucht man eigentlich für die JavaScript Entwicklung. Das läuft alles über NPM. Alle third party libraries holt man über die NPM rein. In der Regel sind das MIT, GNU oder Creative Commons Lizenzen. Gut es gibt auch viele Packages, welche gar keine Lizenz haben. Da gibt es keine Guideline, dass wir die Lizenzen anschauen würden oder so. Da achten wir nicht darauf.

I: Unter welchen Lizenzen werden ihre Projekte veröffentlicht?

B: Also sie werden eigentlich von Anfang an unter einer Creative Commons Share-Alike Commercial veröffentlicht. Ich muss schnell schauen, was die genaue Abkürzung ist. Creative Commons Namensgebung, Weitergabe unter gleichen Bedingungen. Also man kann es weiterverbreiten, transformieren, umwandeln und es kommerziell verwenden. Das ist eigentlich eine sehr offene Lizenz, ausser man muss es danach unter der gleichen Lizenz weiterverwenden. Und wenn man es verwendet, muss man uns als Quelle nennen.

I: Also alles wird unter dieser Lizenz veröffentlicht?

B: Ja.

I: Wie habt ihr diese Lizenz ausgewählt? Wisst ihr das noch?

B: Das ist halt auch schon relativ lange her. Creative Common ist eine relativ klar verständliche Lizenz. Man kann sie konfigurieren, man kann sagen, ob man es kommerziell oder nicht kommerziell will. Es gibt glaube ich auch so einen Konfigurator. Diese Einfachheit war glaube ich der Grund gewesen. Wir sind dann mit dieser Lizenz auch zu unserem Rechtsdienst und der Rechtsdienst war auch der Meinung, dass diese Lizenz Sinn macht.

I: Wie stellen sie sicher, dass die Lizenz von anderen eingehalten wird?

B: Das stellen wir nicht sicher. Da haben wir keine Kontrolle. Das würde unsere Ressourcen sprengen.

I: Also basiert das Ganze eigentlich auf Vertrauen, dass sie zum Beispiel jeweils auch genannt werdet?

B: Ja.

I: Kennen sie Beispiele, in welchen sie genannt wurdet?

B: Man muss grundsätzlich sagen, dass es nicht Dutzende Projekte gäbe, welche unsere Sachen verwenden. Gibt ein paar wenige, von denen wir davon wissen und die haben uns eigentlich immer alle zitiert. Also dort hat es immer funktioniert. Aber es ist sowieso ein fliessender Übergang. Es kann sein, dass jemand unseren Code sieht, Snippets daraus kopiert, Sachen abschaut und es dann nicht gerade eins zu eins zitiert. Dann ist es die Frage, ob dies legitim ist oder ein Bruch der Lizenz.

I: Es erstaunt mich, dass ihr nur 20 Datensätze von Projekten in 3 Jahren auf der Plattform veröffentlicht habt. SRF produziert ja so viel Reportagen, wie zum Beispiel bei für die Rundschau oder 10 vor 10. Sollte da nicht mehr Daten zur Verfügung stehen?

B: Ja, also wir sind eigentlich die einzigen, welche Datenanalyse, also Datenanalyse für journalistische Zwecken machen. Darum denke ich, eignet es sich bei uns auch besonders. Die ganze Veröffentlichung des SRF Materials als open Content oder Shared Content, wie sie dem auch sagen. Das ist eine politische Diskussion. Da gibt es wahrscheinlich auch Überlegungen dazu, da kann ich mich jetzt nicht dazu äussern, weil ich nicht so viel darüber weiss, was da der aktuelle Stand ist.

I: Also, wenn Statistiken dahinterstehen, kommen sie eigentlich in das Spiel?

B: Nein wir kommen in das Spiel, wenn es um Sachen geht, was wir selbst veröffentlichen. Aber man muss natürlich auch sehen, die Artikel, welche wir dazu schreiben und auf Srf.ch publizieren, irgendwelche Radiogespräche oder Fernsehbeiträge, die stehen nicht unter einer Creative Commons Lizenz. Die sind unter dem gleichen Rechtekonsturkt wie alle andere SRF Publikationen.

I: Unter welcher sind die?

B: Das weiss ich jetzt auch nicht auswendig. Aber das ist grundsätzlich Urheberrecht. Und vor allem unser Code, das ist eigentlich auch Urheberrecht. Aber dort heben wir das ja durch die Lizenzen auf. Aber ich muss ehrlich sagen, bin auch kein Jurist, was mit dem Urheberrecht passiert. Ob es jetzt ein Urheberrechtsverstoss wäre, wenn man den Code nimmt und nicht zitiert. Das weiss ich jetzt nicht.

I: Es ist also eine politische Frage? Es gibt geschäftskritische Inhalte, welche zum Beispiel ein Wettbewerber eins zu eins übernehmen könnte und die schützt ihr. Und dann gibt es geschäftsunkritische Inhalte wie zum Beispiel ein Datensatz, welchen man zuerst verstehen, aufbereiten und vielleicht noch einen Bericht darüberschreiben muss. Und diese veröffentlicht ihr?

B: Genau. Wegen dem geschäftsunkritischen. Wenn wir die Daten veröffentlichen, dann besteht nicht gross ein Wettbewerbsnachtteil, weil grundsätzlich haben wir unsere Story schon gemacht. Es kann sein, dass jemand ein Jahr später, wenn es neue Daten gibt, diesen Code nimmt, mit neuen Daten aufbereiten und selber eine Story macht. Kommt relativ wenig vor und wenn es vorkommt, ist es auch nicht schlimm. Das ist ja eigentlich ja auch das, was wir auch wollen. Wir wollen ja anderen Datenjournalisten dabei helfen, auf dieser Arbeit aufzubauen.

I: Noch eine Frage zu Schulungen. Werden eure Mitarbeiter mit dem Umgang auf GitHub, Lizenzen oder generell Open Source sensibilisiert und geschult?

B: Jaja das sicher. Also jeder, der bei uns arbeitet, arbeitet sowieso mit GitHub und weiss dementsprechend damit umzugehen. Und wenn Redaktionsgäste oder Praktikanten kommen, dann werden die am Anfang eigentlich immer auf dieses System geschult.

I: Gibt es ein offizielles Training?

B: Nein, das ist nicht offiziell. Das ist auch wieder informell, einfach in unserem Team.

I: Und wie ist das beim Online Team, wisst ihr das?

B: Da kann ich nichts dazu sagen.

I: Kommen wir zum Thema Partner. Ich habe auf GitHub gesehen, dass ihr ein paar Datensätze mit dem Namen „interaktiver Journalismus“ publiziert habt. Mit Aljazeera oder Berlinerpost.

B: Ja das ist eigentlich nur eine Liste. Es gibt zum Teil Kompilationen und Listen auf GitHub, wo es eigentlich darum geht aufzuzeigen, was Datenjournalisten im interaktiven Bereich gemacht haben oder als Recherche.

I: Habt ihr jetzt dort mitgearbeitet? Zum Beispiel die Kartenvisualisierung mit den syrischen Flüchtlingen habe ich mir angeschaut, dass man irgendwo auf der Welt draufklicken kann und man sieht, wieviel Fläche sie benötigen würden.

B: Nein, da haben wir nicht mitgearbeitet. Ich weiss nicht genau was ihr meint, aber ich glaube sie sprechen von einer Liste mit verschiedenen Projekten?

I: Genau.

B: Wir haben dort bei dieser Liste eigentlich nur unsere Sachen hinzugefügt, also verlinkt. Mehr haben wir da nicht gemacht.

I: Arbeitet ihr mit Partnern zusammen?

B: Nein, also die einzigen Partner wo man zusammen auf GitHub entwickelt, sind

eigentlich Firmen, welche uns bei der Datenvisualisierung helfen. (...). Ihre Codes sind dann auch auf GitHub, aber es sind nicht die Codes, welche wir veröffentlichen. Weil Frontend Codes veröffentlichen wir grundsätzlich nicht, sondern nur Analysencodes.

I: Wieso veröffentlichen Sie die Frontendcodes nicht?

B: Ja, grundsätzlich finden wir, dass es dort weniger relevant ist. Da geht es nicht um die Auswertung und Analysen von Daten. Es geht nicht um die Gewinnung von Rechercheergebnisse. Darum finden wir, dass man da nicht sehr transparent sein muss, wie bei den Auswertungen. Das ist eigentlich so der Hauptgrund. (...). Und ein stückweit sicher auch noch eine Art Geschäftsgeheimnis, welches wir bewahren wollen.

I: Also ein Alleinstellungsmerkmal?

B: Ja. Die Frontendentwicklung ist halt auch aufwendig und ja da haben wir entschieden, dass das unser Geschäftsgeheimnis ist, wie wir unsere Applikationen bauen.

I: Wie habt ihr eure externen Partner für die Visualisierung ausgewählt?

B: Ja ist einfach ein normaler Beschaffungsprozess. Dann schaut man, wer kann die Anforderungen erfüllen und danach ist es ein complianceorientierter Beschaffungsprozess. Aber es ist natürlich schon so, dass wenn man mit paar Firmen positive Erfahrungen gemacht hat, kann man auch argumentieren, dass man mit ihnen zusammenarbeiten will. Vor allem auch, weil schon Vorwissen vorhanden ist und dann arbeitet man oft mit ähnlichen Firmen zusammen.

I: Ist es auch ein Kriterium bei der Auswahl, dass diese Firmen auf Open Source setzten?

B: Dass sie selbst Open Source verwenden?

I: Genau

B: Das ist kein Kriterium. Denn es muss auch kein Kriterium sein, weil in der Softwareentwicklung bzw. in der Frontendentwicklung, ist es sowieso Standard. Es geht gar nicht mehr ohne Open Source.

I: Ab welchem Zeitpunkt involviert ihr Partner bei euren Projekten?

B: Das ist unterschiedlich. Zum Teil in der Designphase, wenn es darum geht, Inputs zu haben, wie man etwas designen könnten oder wie man die Usability machen könnte. Vielleicht auch nur für das Design. Vielleicht auch nur in der Entwicklung selbst. Das ist völlig unterschiedlich.

I: Ihr habt vorher die Open Data Community angesprochen. Arbeitet ihr mit Communities zusammen?

B: Ja, relativ wenig. Ich sage mal, dass die Verbindungen eher informell sind, weil man sich schon gegenseitig kennt. Wir waren auch schon an Hackathons, vielleicht heutzutage weniger als früher. Auch aus Ressourcengründen wiederum. Was ich weiss, dass an vielen Hackathons auf die Daten, welche wir veröffentlichen, verwiesen wird. Abgesehen von dem gibt es nicht riesige Berührungspunkte.

I: Seid ihr offiziell Teil einer Community?

B: Kommt darauf an, wie man Community definiert. Wir haben einen Twitter Community, welche uns folgt. Wir folgen anderen.

I: Seid ihr Teil von der Open Data Community?

B: Ich würde uns schon als Teil bezeichnen, wir als Privatpersonen, auch weil man sich kennt. Aber wir sind nicht hochoffiziell vertreten in einem Vorstand oder was auch immer.

I: Erwartet ihr jetzt Beiträge von einer Community für eure Arbeiten?

B: Nein.

I: Ihr entwickelt ja auch alles selbst intern.

B: Ja, da erwarten wir nichts. Was natürlich vorkommt, dass wenn man gewisse Kontakte in dieser Szene hat, ruft man vielleicht jemanden an und fragt. "Weisst du ob es solche Daten gibt, oder wo ich die erhalten könnte?" Also das informelle Netzwerk ist schon hilfreich. Aber es ist jetzt keine Erwartungshaltung da. Auch gegenseitig glaube ich auch nicht, dass die eine Erwartung gegenüber uns haben.

I: Dann betreiben sie auch kein aktives Community Management?

B: Nein das nicht. Aber auch da muss man unsere Ressourcen im Auge haben. Wir sind

auch nur drei Leute. Das könnte wir uns gar nicht leisten so etwas.

I: Wie gehen sie mit der Imitationsgefahr um? Wenn jetzt zum Beispiel jemand den gleichen Artikel aus ihren Daten schreiben würden?

B: Ja, also was halt im Journalismus vorkommt, aber relativ selten, dass andere Medien vielleicht etwas übernehmen und zum Teil sogar ganze Sätze abschreiben. Das ist halt eine journalistische Unart. Das hat jetzt nicht mit Open Source und Code und was auch immer zu tun. Zum Teil schreibt man vielleicht auch ein Email. Ein fiktives Beispiel wäre, wenn die NZZ schreiben würde: "Der Tagesanzeiger hat herausgefunden, dass..." und dabei wären wir es gewesen, welche es heraufgefunden haben. Dann schreiben wir ihnen vielleicht ein Email, aber kommt jetzt sehr selten vor.

I: Und jetzt zum Beispiel die Datensätze und Visualisierung? Ihr habt vorher erwähnt, dass Frontendcodes nicht veröffentlicht werden. Wenn sie zum Beispiel sehen, dass jemand das gleiche hat, wie geht ihr vor?

B: Das ist noch nie vorgekommen. Wenn es vorkommen würde, dann würden wir das mit dem Rechtsdienst abklären. Denn das ist dann schon eine Urheberverletzung.

I: Erhaltet ihr Feedbacks von anderen Leuten oder werdet ihr auf Fehler hingewiesen, wenn sie etwas veröffentlichen? Ich habe gesehen, dass ihr auf GitHub einen Satz habt, dass ihr offen für Feedbacks seid.

B: Auch da sind Feedbacks relativ klein. Wie schon gesagt, es sind nicht tausende von Leuten, welche unsere Daten herunterladen und laufen lassen. Und konkrete Fehler oder so, das ist noch fast noch nie vorgekommen. Was auch vorkommt, aber auch sehr selten, dass jemand nur den Onlineartikel liest und danach findet, dass es eine komische Annahme ist, welche wir getroffen habt. Dann schreiben wir jeweils zurück und versuchen uns zu erklären. Da kann man dann oft auch auf den Code verweisen, schwarz auf weiss aufzeigen wie wir vorgegangen sind und als Diskussionsbasis brauchen.

I: Bei Onlineartikel von ihnen ist mir nie aufgefallen, dass es einen Hinweis zum Code hat.

B: Dann haben sie einfach nicht gut genug geschaut. (lacht). In der Regel wird das schon verlinkt.

I: Unter dem Artikel?

B: In der Regel, ja. Ja, wir haben in der Regel eine kleine Box, welche wir die Methodik für Laien erklären.

I: Die habe ich schon gesehen.

B: Danach ist in der Regel ein Link mit: "Mehr dazu finden sie auf dem Open Data Portal von oder finden sie hier". Das sollte schon die Regel sein.

I: Aber nur ist halt nur auf die Artikel von ihnen?

B**:** Ja nur bei unseren Artikeln.

I: Finde es sehr interessant, dass jetzt SRF Open Source veröffentlicht. Den Code vielleicht nicht für ein Frontend- oder Softwarelösung, sondern eher ihr Wissen in Form von Datensätze teilt, wo andere darauf aufbauen könne. In unserer Studie geht es eben um neue Akteure in der Open Source Bewegung zu identifizieren, die Motivationsgründe, Geschäftsfälle, Richtlinien und Compliance zu analysieren. Wir werden die Arbeit auf der Homepage der Digitalen Nachhaltigkeit von der Universität Bern veröffentlicht. Wenn das in Ordnung ist

B: Ja, das war mir schon bewusst. Aber was man sehen muss, SRF hat über 2000 Mitarbeiter und wir sind drei davon. Wir sind so ein kleiner Teil. Wir publizieren etwas in dieser Nische des Datenjournalismus. Es ist nicht so, dass SRF per se eine riesen Open Source Strategie hat oder so, aber das ist glaube ich auch klar geworden. Das muss man einfach sehen.

I: Ähnliches haben wir bei anderen Firmen, welche nicht aus der IT stammen, beobachtet, dass es meistens Bottom-up kommt.

B: Genau.

I: Aber vielleicht könnt ihr ja diesen Geist ein bisschen bei den anderen verbreiten. Und das vielleicht auch eine grössere Öffnung bewirken könnte.

B: Was man sicher sagen kann, dass unsere Arbeit andere Journalisten inspiriert hat, das ähnlich zu machen. Vor allem natürlich im deutschsprachigen Raum. Wir sind nicht die Einzigen. Zum Beispiel bei der Sonntagszeitung, der eine Journalist Barnaby Skinner, der veröffentlicht ab und zu Python Codes. Oder zum Beispiel bei dem Bayerischer Rundfunk, die haben ein ähnliches Team wie unseres und haben auch angefangen. Es gibt Kollegen aus Österreich, welche unser mit unserem Template arbeiten. Da haben wir glaube ich schon etwas in Bewegung gesetzt. Da sind wir auch nicht die Einzigen, welche das machen.

I: Wurdet ihr von anderen inspiriert, dies zu tun?

B: Was man sagen muss, grundsätzlich im Datenjournalismus, was ja eher neuere Erscheinung ist. Das Ganze ist schon vom angelsächsischen Raum inspiriert. Zum Beispiel New York Times und alle diese grossen Players. Man muss aber auch sagen, zum Beispiel die FiveThirtyEight, das ist so eine Plattform in Amerika, welche nur Datenjournalismus macht. Die gibt es seit 2014 oder so. Und die haben am Anfang ab und zu, nicht ihren Code, aber Daten, welche sie aufbereitet haben, auf GitHub veröffentlicht. Alles zusammen in einem einzigen Repository. Ich kann mich nicht gross daran erinnern, aber das könnte schon eine Art Inspiration gewesen sein für uns. Wir haben uns aber auch gesagt, wenn wir es machen, dann sicher nicht so. Denn einerseits wurde kein Code veröffentlicht, anderseits ist es nicht wirklich reproduzierbar und gut dokumentiert. Ich bin der Meinung, dass wir danach relativ schnell eine höhere Messlatte gelegt haben, als viele von diesen amerikanischen grossen Medien.

I: Sie setzten eher auf Qualität statt Quantität?

B: Ja das sowieso. Gut, wir können auch nicht so auf Quantität setzten, wegen unserer kleinen Grösse. Uns ist es wirklich wichtig, dass wenn wir Sachen veröffentlichen, die nachvollzierbar und gut dokumentiert sind, damit jemand mit Grundkenntnissen im Programmieren etwas damit anfangen kann. Dass es auch fehlerfrei funktioniert.

I: Ich habe noch eine Frage über Journalismus, weil ich selbst nicht so tief drin bin. Ihr habt immer wieder Datenjournalismus erwähnt. Wie kann man das vom anderen Journalismus abgrenzen?

B: Das kann man eben nicht gut abgrenzen. Die Grenzen sind sehr fliessend. Es gibt auch verschiedene Definitionen was Datenjournalismus ist und was nicht. Es gibt Leute, die sagen. dass es erst Datenjournalismus sei, wenn man mindestens 100 Gigabyte verarbeitet. Ja es gibt verschiedene Definitionen. Meine Definition ist, sobald man schneller ist, wenn man Sachen automatisiert, als wenn man nicht automatisiert, dann ist man datenjournalistisch unterwegs. Ist sehr eine sehr abstrakte Definition. (...) Sie verhebt nicht 100 prozentig. Man kann auch Sachen mit einem Taschenrechner ausrechnen und dann ist man vielleicht auch noch datenjournalistisch unterwegs, weil man Daten zusammenträgt. Es gibt so viel verschiedene Spielarten von Datenjournalismus. Datenjournalismus kann zum Beispiel auch die Panamapaper sein. Die haben Terabytes von Leaks zur Verfügung gehabt. Diese Leaks nach einem einzigen Wort zu durchsuchen, wäre schon Datenjournalismus. Denn man muss da viel Engineering betreiben, dass man das Leak gescheit durchsuchen kann. Man muss vielleicht einen Index bauen und so weiter. Da braucht es auch schon technische Hilfsmittel und eigentlich automatisiert man dann schon. Für Datenjournalismus muss man nicht unbedingt eine Tabelle auswerten per se. Aber man kann auch eigentlich schon eine Summe aus einer Tabelle mit 26 Zeilen für jeden Kanton bilden und das kann man auch schon als Datenjournalismus definieren.

I: Mich überrascht, dass in unserem digitalen Zeitalter von 2000 Mitarbeiter nur 3 Vollzeit Datenjournalismus betreiben.

B: Ja, man muss aufpassen. Wir haben dieses Team einfach institutionalisiert. Aber es gibt natürlich viele Mitarbeiter bei der SRF, welche mit Excel Auswertungen machen. Es gibt auch zwei, drei andere, welche ein bisschen coden können und vielleicht auch so etwas machen. Wir haben jetzt auch mehr Leute in der Ausbildung. Jetzt gibt es den Diplomstudiengang am MAZ in Luzern.

I: Denken sie der Journalismus wird sich in Richtung Datenjournalismus verschieben?

B: Nicht unbedingt. Ich kann jetzt nicht prophezeien, dass in paar Jahren viel mehr Leute Datenjournalismus machen werden. Solche Prognosen sind sowieso extrem schwer. Wird Datenjournalismus eine Nische bleiben, denn heute ist es eine Nische oder wird es so

sein, dass jeder zweite Redaktor Daten auswerten wird. Das ist schwierig zu sagen.

I: Sie haben gesagt, dass sie als Quereinsteiger angefangen haben, oder?

B: Ich schon ja. Aber es gibt auch Leute wie zum Beispiel mein Kollege Julian Schmidli, welche effektiv Journalismus studiert haben. Aber ich denke Datenjournalismus ist für viel nicht Journalismus eine gute Einstiegsmöglichkeit, weil es halt auch andere Eigenschaften wie technische Skills braucht. Diese Zeit ist zwar wieder ein wenig vorbei, es hat sich ein bisschen gesättigt. Als ich angefangen habe, waren Leute mit Programmier- und Allgemeinbildung sehr gefragt und darum bin ich relativ schnell nach meinem Studium in diesen Journalismus reingekommen.

I: So wie ich es verstanden habe, coded ihr nicht nur, sondern schreibt auch den Artikel dazu?

B: Ja, ich mache das ganze Paket.

I: Macht immer eine Person ein Paket?

B: Nein, zum Teil arbeiten wir auch im Team zusammen, dass wir Aufgaben aufteilen, das ist ja klar. Ich sage mal jeder von uns kann Artikeln schreiben, kann recherchieren und Codes schreiben.

I: Dann habt ihr nicht irgendwelche Rollenverteilung für Datenaufbereitung oder Verfassen. Jeder kann in dem Fall alles?

B: Ja, klar hat man zum Teil Stärken und Schwächen.

I: Aber von der Jobbeschreibung her gibt es nicht jemanden für die Datenaufbereitung, Datenvisualisierung oder ein Redaktor.

B: Nein. Das mag in gewissen Jobbeschreibung so sein, dass ein Team bewusst einen Designer sucht, aber oft zeigt sich das die Leute je nach Stärken und Schwächen andere Aufgaben übernehmen. Datenjournalismus ist in dem Sinn ein sehr kreatives Feld, in welchen man nach seinen Möglichkeiten weiterentwickeln und andere Aufgabe übernehmen kann. Das ist wirklich auch etwas Großartiges an diesem Job.

I: Habt ihr im Team eine flache Hierarchie oder ist jemand der Chef?

B: Nein, wir haben eine sehr flache Hierarchie oder haben auch keinen offiziellen Chef im Team selbst. Natürlich untersehen wir Vorgesetzten von weiter oben. Aber bei unserem Team ist es flach. Aber es ist natürlich so, jedes Projekt hat einen Leiter, eine klassische Projektleitung, Projektmanagement. Schwierige Entscheidungen oder so die treffen wir sowieso immer im Team.

I: Ihr reportet dann noch zu jemand oberhalb.

B: Genau.
I: Vielen Dank für Ihre Zeit.

Interview mit Markus Tiede am 16.01.2019

I: Zuerst noch wegen den Daten. Also die Arbeit wird auf Matthias Homepage für Digitale Nachhaltigkeit publiziert. Das Interview wird dann als Transkript im Anhang sein und gewisse Teile werden als Zitate oder Verweise in der Arbeit erscheinen.

B: Mhm (bejahend). Perfekt. Das ist dann eine Masterarbeit, oder?

I: Ja genau in Form einer Masterarbeit. Dann zuerst mal die einleitenden Fragen über sie persönlich. Was ist ihre Ausbildung, wie sind Sie zur Baloise gekommen und was ist ihr momentane Funktion?

B: Mein Name ist Markus Tiede. Ich bin Softwareentwickler seit 10 Jahren. Ich habe Informatik studiert in Deutschland an der Fachhochschule in Wolfenbüttel, Braunschweig. Ich habe dann lange Zeit in Braunschweig gearbeitet. Für ein Unternehmen, welches Quelloffene Software bei der Eclipse Foundation entwickelt hat. Ich bin dann über einen kleinen Umweg auf die Basler Versicherung aufmerksam geworden. Ich habe zu den Basler vor zwei Jahren gewechselt, ende 2016. Und mitunter ausschlaggebend war der Faktor, dass die Basler als eine Versicherungsgesellschaft einen eigenen GitHub Account hatte. Und eigene Open Source Projekte hatte. Und ich habe gedacht in diesem Unternehmen scheint es etwas anderes zu laufen als bei anderen Versicherern. Weil andere Versicherungen hatte ich so noch gar nicht wahrgenommen. Ich habe dann bei der Basler begonnen. Nicht direkt als Softwareingenieur, sondern als Release Manager. Ich habe bei unserem Grossprojekt eineinhalb Jahre lang als Releases betreut. Und jetzt vor einem halben Jahr habe ich einen temporären Jobwechsel angenommen. Dieser Jobwechsel ging genau um das Thema Open Source Entwicklung bei der Baloise Group. Der Jobwechsel ist im Dezember ausgelaufen und jetzt bin ich im Integration Services Team, die bieten so querschnittliche Dienstleistungen für alle anderen Teams bei der Basler an. Infrastrukturdienste.

I: Diese Dienste sind nicht nur auf Open Source ausgerichtet?

B: Genau, das sind mitunter Open Source Produkte von Red Hat die wir einsetzten, zum Beispiel Red Hat Fuse als Enterprise Service Bus den wir zurzeit einführen. (...) Aber maximal use von Open Source Software aktuell. Was jetzt kommt ist das Thema API Management, auch für unser Team. API Management für die Basler aufzubauen im laufen dieses Jahrs. Und das ist ein Feld, wo wir eben sehen, dass Open Source Teile sehr stark beim API Management helfen könnten. API bekannt zu machen und zu verbreiten.

I: Wie ich schon erwähnt habe wir für unsere Studie Unternehmen und Behörden gesucht, welche nicht aus der IT stammen, aber anfangen Open Source Software selbst auf GitHub veröffentlichen. Zuerst einmal, wie entstand die Idee bei Baloise eigene Projekte, Software oder Teile der Software zu veröffentlichen und der Allgemeinheit quasi kostenlos zur Verfügung zu stellen?

B: Also die Idee selbst habe ich in den Anfängen nicht begleitet, das war Matthias Cullmann, den ich da federführend wahrgenommen habe. (...) Soweit ich das eben wiedergeben kann, kam das Ganze im Zuge der Agilisierung. Also ich nenn das mal Agilisierung. Vor fünf bis sechs Jahren sind bei uns Querschnittlich, sogenannte Competence Center etabliert worden. Das heisst zu den einzelnen Scrum Teams quasi Experten Gruppen, die teamübergreifend sich austauschen konnten. Und dort gibt es eine Gruppe zum Beispiel für Architektur, eine Gruppe für Qualitätssicherung, eine Gruppe für Software Engineering. Und im Kontext der Software Engineering Gruppe ist dieses Thema eben aufgenommen worden. Und in dem Kontext gab es dann schon seit Anfang an das Konzept von sogenannten Goldcards. Das ist das wir ein bisschen von Google kopiert haben. Die meisten Teams schaffen in zwei Wochen Sprints. Und innerhalb dieser 2 Wochen kann sich jeder Sprintteilnehmer einen halben Tag lang Zeit nehmen Dinge zu tun, so genannte Goldcards. Die für das Team Nutzen stiften. In den Anfängen sind viele dieser Goldcards Ideen und Projekte eben einfach auf GitHub gestellt worden. Kleine Codes Snippet die in dieser Zeit entstanden sind, die man dann veröffentlicht hat.

I: Und die Mitarbeiter hatten alle Freiheiten für diesen halben Tag und konnten machen was sie wollten?

B: Genau. Also es wurde im Anschluss an den Sprint, dem Team vorgestellt. Also musste man

dann eigentlich sagen, ok ich habe das gemacht und das Ganze war auch irgendwie sinnvoll. Man kann sich genauso gut etwas anschauen, eine neue Bibliothek und die dann vorstellen oder solche Dinge. Manchmal sind es eben dann Summen von Goldcards, dass sich mehrere Leute sich zusammengetan haben und genutzt, um kleine bis mittelgrosse Projekte zu realisieren.

I: Und so haben gewisse Leute Open Source als Thema aufgegriffen, analysiert und überlegt wie man das bei der Baloise nutzen kann.

B: Genau. Vielleicht sprechen wir später noch über das Maturitätsmodell, das wir verwenden, welches initial von der Eclipse Fundation stammt. Im Prinzip befindet man sich damit in Bezug auf Open Source Software in einem weiteren Stadium der Adaption. Man versteht, dass selbst die Freigabe von kleinen Komponenten Sinn machen kann und die Attraktivität sowohl für Mitarbeiter als auch Aussenstehende darstellen kann. Ich glaube das Unternehmen hat schon lange auf Open Source Software gesetzt. Also diese erste Stufe des Maturitätsmodells die von Deny weggeht, ist dass man einfach anfängt Open Source Software zu nutzen. Das ist bei der Basler Versicherung schon lange genutzt worden.

I: Was für Projekte, Software oder Codes haben sie bereits unter einer Open Source Lizenz entwickelt und veröffentlicht oder planen sie zu veröffentlichen?

B: Also das ist eben sagen wir mal relativ divers. Also von den aktuellen Statistiken auf GitHub haben wir ungefähr 60 Leute. Bevor ich auf die Projekte eingehe gehe ich zuerst auf die Personen, die involviert sind, ein. Ungefähr 62 Personen, die bei der Basler Teil der GitHub Organisation sind. Und von diesen 62 Leuten sind vielleicht fünf bis zehn mittelmässig aktiv. Mittelmässig aktiv deswegen, weil wir bislang eben Projekte veröffentlicht haben, die aus dem Kontext solcher Goldcards Arbeitsprojekte entstanden sind. (...). Ich glaube, dass die Anzahl Leute ungefähr auch Anzahl Projekte entspricht. So an die 50 Projekte, die einen ganz unterschiedlichen Grad der Maturität haben. Also manche Projekte sind einzelne Helperklassen, wo man auf Sourcecode Ebene eine Wiederverwendbarkeit sehen kann, die aber selber zum Beispiel nicht releast oder verteilt werden. (...). Das ist so die eine Klasse von Projekten. Die nächsten Klassen von Projekten, die gerade in den letzten Jahren dazu kamen, waren Repository / Projekte, die eigentlich so etwas wie Workshop, Fortbildung oder Schulung bei der Basler darstellen. Also wenn wir interne Schulungsunterlagen vorbereiten oder aufbereiten, machen wir das mittlerweile auf GitHub. Ich würde mal sagen, da haben wir drei bis vier mehrtägige Workshops, die auf GitHub stehen. Die in der Form zwar releast werden, aber nur im Sinne von, das sind Folien und Beispiele, die man vor einem Publikum vortragen kann. (...). Und dann gibt es vielleicht so zwischen fünf und zehn Projekte, die man wirklich vielleicht als Projekt wahrnehmen würde, weil sie extern aktiv konsumiert werden. Zum Beispiel auch schon bei den Workshops, haben wir das auch, dass die geforkt werden und extern gehalten werden. Das nehmen wir schon wahr. Aber bei den anderen, zum Beispiel haben wir zwei Plugins für Atlassian Produkte. Beides Plug-ins für Confluence. Also wir nutzen Confluence gurppenweit ziemlich intensiv. Und haben einen Plug-in zum Anfertigen von rechtswirksamen digitalen Unterschriften im Unternehmen, das heisst digital signature. Das hat so ungefähr 300 Installationen weltweit. Und ähnlich sieht es aus mit einem Plug-in, das heisst Dashboard plus. Das kann zum Beispiel Unit Test Ergebnisse visualisieren. Das wird auch regelmässig releast, von mehreren Leuten mainaint und weltweit bei ungefähr 300 Installationen konsumiert. Also das würde ich sagen sind so die verschiedenen Grössen an Projekten, die wir haben. Wir haben noch kein echtes strategisches internes Projekt, was komplett quelloffen entwickelt wird. Das sind eher Hilfsprojekte, die bei uns im Rande der normalen Softwareentwicklung entstanden sind.

I: Du hast vorher angesprochen, dass über die Goldcards alles eigentlich initiiert wurde.

B: Vieles, ja.

I: Könnte man da von einem Button-up Ansatz sprechen?

B: Ja definitiv. Also es ist aktuell noch nicht so, dass es vom Management vorgegeben wird. Das ist der Weg, den ich versucht habe zu bereiten im letzten halben Jahr.

I: Und wie war deine Erfahrung mit dem Management?

B: (...) Sehr positiv eigentlich. Weil (...). Bestimmte Leute, also so wie du das vorhin auch schon beschrieben hast, Open Source findet nicht nur bei IT Firmen Anklang. Und es wird nicht nur von IT Firmen in der operativen Ebene, sondern es wird von oben als Chance erkannt. Bei der Basler ist es so, dass es im Management schon in bestimmten Bereichen erkannt wird. In bestimmten Bereichen sind keine Überzeugungsleistung notwendig und an anderen Stellen fällt die Überzeugungsleistung viel einfacher, weil es im Management schon Leute gibt, die das Ganze auf gleicher Ebene transportieren können.

I: Und wo habt ihr Schwierigkeiten oder wo braucht es noch Überzeugungskraft?

B: (...) Ich würde fast sagen, mehr bei den Mitarbeitern. Mehr bei den Mitarbeitern, bei den Teams. Wir sind ungefähr 150 bis 200 Softwareentwickler

I: In der Schweiz?

B: Ja hier am Standort Basel. Also das ist zum Teil Schweiz zum Teil Gruppen IT und von diesen 150 bis 200 Mitarbeitern, also auch externe Dienstleister, die bei uns tätigen sind, können wenn sie wollen, dort teilnehmen. Manche haben das auch sehr erfolgreich gemacht, vielleicht gehen wir danach noch darauf ein. (...) Aber eben nur sagen wir mal 10 bis 50 dieser 150 bis 200 Leute sind aktiv irgendwie schon mal mit Open Source Community in Berührung gekommen. Was ich sagen möchte, das Management hat es eigentlich verstanden, denn viele neue Projekte, die wir starten wollen, wollen wir auch grösserenteil quelloffen starten. Aber die Leute mitzunehmen das ist aktuell die grössere Herausforderung, als das Management davon zu überzeugen.

I: Habt ihr gewisse Manager, die den Kopf hinhalten und Open Source Software aktiv pushen?

B: Ja genau. Also den Kopf hinhalten, ich würde mal sagen, da sind wir noch nicht. Aber die sich dafür stark einsetzten, ja. Denn wir haben noch keine Projekte, jetzt einfach so vom Lifecycle eines Projektes, wo man sagen müsste, man müsste ein Projekt gross verteidigen, weil es in Produktion ist. Und vielleicht Mitbewerbern auch einen bestimmten Vorteil verschafft, indem eigenen Code quelloffen veröffentlicht. Zum Beispiel ein Kernsystem der Versicherungsbranche. Also da sind wir noch nicht ganz, dass man das stark verteidigen müsste.

I: Ich habe auf dem GitHub Account der Baloise Open Insurance und Open Prevo entdeckt. Können sie dazu noch etwas sagen?

B: Ja kann ich gerne etwas dazu sagen. Also wir haben versucht eine relativ klare Terminologie zu verwenden, wenn wir über Open Source sprechen. Ähnlich wie das zu Red Hat auch zum Beispiel tut. Aktuell passiert bei uns alles auf GitHub und von daher lehnen wir einen grossteil der Terminologie natürlich einfach an, wie GitHub es nennt. Auf GitHub gibt es quasi Projekte und Organisationen und die Organisation für die Basler Gruppe, die gibt es schon relativ lang, ungefähr vier bis fünf Jahren. Es ist der Container der Entwicklung, die wir sehen, dass es Software die direkt von der Basler für die Basler geschrieben wurde, die vielleicht Synergien auch woanders darstellen kann. (...). Jetzt habe ich in meiner Tätigkeit vor der Basler schon häufig erlebt, dass man eigentlich, wenn man in einem grösseren Stil zusammenarbeiten möchte, einen neutralen Grund braucht, auf dem man sich bewegt. Und dieser neutrale Grund, und deshalb ist der Zustand von Open Insurance auch Pre-Proposal. Aktuell ist es in einer Phase, wo wir einen Vorschlag haben, wir das ganze Thema diskutieren oder wo es auch weltweit diskutiert wird. Was könnte sich hinter dem Trend von Open Insurance verbergen. Und wir haben einfach angefangen den Begriff zu benutzten und zu überlegen, was bräuchte es, um auf dieser Ebene agieren zu können. Klar man bräuchte zuerst einmal ein Wording und man bräuchte erst mal ein Framing. Man bräuchte vielleicht eine Gruppe an Leuten, die dieses Mindset haben. Und da sind wir aktuell dabei diese Leute anzuschreiben, aber nicht wirklich aktiv. Also da ist noch relativ wenig im Gang. Aber das ist die Vision von Aktivitäten einer einzelnen Versicherung auf neutralen Grund zu heben. Und nicht zu sagen, das ist primär von der Basler gesteuert, sondern das ist ein Verbund von Versicherern, die an Quelloffener Software interessiert sind, auf all diesen Ebene, die es vom Maturitätsmodell gibt.

I: Also das heisst Zusammenarbeit mit der Konkurrenz, wenn ich das richtig verstehe?

B: Genau. Anstoss für das gesamte Denken richtig Open Insurance war ein Schritt der

Allianz Deutschland im Dezember 2017. Da ist auf C-Level Ebene kommuniziert worden, dass die Allianz ihr komplettes Kernsystem quelloffen machen wird. Und das hat relativ weite Wellen geschlagen, so vom Mindset her wachgerüttelt. Hei, die Allianz macht da etwas, das ist für die Branche unüblich. Im Jahr 2018 haben wir davon noch nicht viel gesehen. Wir treffen uns im nächsten Monat mit den Cs und sprechen darüber, aber das war der Stein des Anstosses. Das war Ende 2017. Am Anfang haben wir angefangen diesen Term einfach zu nutzten. Da sind zwei weitere Bewegung dazu gekommen. Eine Open Insurance Initiative OIIN oder OIN. Ansässig laut GitHub in Abu Dhabi. Und eine Bezwingung in Köln, die heisst Open Cologne hat aber dieselbe Vision. Bei Open Cologne steckt die AXA dahinter, mit noch zehn bis 15 anderen deutschen Versicherungen. Was in der Open Insurance Initiative steckt kann ich noch nicht ganz beurteilen. Aber man merkt das Mindset ist irgendwie da, man versucht zurzeit auszuloten was für Projekte man sich hinter diesem Deckmantel vorstellen könnte. Diese einheitliche Sicht auf die Nutzung oder Freigabe von Open Source Software.

I: Könnt ihr eine Analogie zu einer anderen Industrie machen. Gibt es so etwas ähnliches wie ein neutraler Grund schon?

B: Ja, also wir sprachen vorhin kurz davon, dass wir Mitglied der Eclipse Foundation sind. Und in vielen Not for Profit Open Source Organisation wie zum Beispiel die Eclipse Foundation aber auch Linux Foundation, gibt es häufig Industriearbeitsgruppen, die sich zusammentun, um zu überlegen, wo kann man Synergien schaffen und wo kann man Redundanzen vermeiden, indem man sich eben zusammentut. Und ich habe das eben stark in der Eclipse, weil das mein Hintergrund war, in sechs Jahre quasi indirekt für die Foundation gearbeitet und vielen in dem Kontext mitbekommen. Dort gibt es eine ganze Reihe Industry Working Groups, die das tun. Also die aktuell prominentesten, du bist ja selbst auch bei Bosch tätig, die IOT Working Group. Zum Beispiel bei der Eclipse Foundation, wo Bosch federführend ist, wo aber auch anderen grössere Players an den Tisch kommen und überlegen, wo sie gemeinsam Software schreiben.

I: Also seid ihr jetzt mit dem Pre-Proposal für die Open Insurance federführend?

B: Das ist das was wir uns zurzeit überlegen ist, wo wir uns da positionieren. Wir haben das Business bei uns involviert und haben noch eine Reihe weiterer Initiativen. Weitere Initiative wo es aktuell gibt, ist zum Beispiel die Open Baloise. Irgendwie der Zwischenschritt zwischen Open Insurance und Baloise an sich zum Beispiel. Wir haben zum Beispiel Insurance in a Box als Begriff, da müssen wir jetzt nicht viel tiefer darauf eingehen, aber es sind alles Themen, die eine starke Schnittmenge miteinander besitzen und alle irgendwie dahingehen, dass das Unternehmen sich nach aussen hin öffnet und man es kleiner modularisiert und wiederverwendbar macht.

I: Aber die Open Insurance wurde nicht von der Baloise gegründet?

B: Es gibt keine Rechtsform an der Stelle. Es gibt keine Arbeitsgruppe, keine Rechtsform. Es gibt aktuell eine Website, einen GitHub Account und eine Reihe von Leuten, die Interesse haben über das Thema zu sprechen und sich Gedanken zu machen.

I: Das wurde alles von Baloise initiiert?

B: Ja mit der Abgrenzung, die ich gerade schon getätigt habe. Open Cologne kam von der AXA und Open Insurance Initiative kam nochmals von jemanden anderem. Und dann Open Prevo ist vielleicht auch noch spannend. Open Insurance ist quasi schon die ganz grosse Vision oder die grössere Vision. Viele sagen uns, dass dies sehr schwer zu realisieren, weil viele Regularien, die in der Versicherungsbranche gelten sind sehr Länderspezifisch. Und das ist vielleicht eine ganz gute Überleitung zu Open Prevo. Die Basler entwickelt schon seit vielen Jahren gemeinsam mit der Helvetia und der Zürich Versicherung in der Schweiz das Kollektiv-Leben Versicherungssystem PAKT. Und zwar nicht quelloffen. Ist quasi ein Joint Venture, aber ist nicht quelloffen. Und Open Prevo ist als MVP mit letztem Jahr einmal für 6 Wochen gestartet worden und hatte als Zielsetzung ein Minium Viable Product zu schaffen, mit dem man zeigen kann, dass man so etwas wie eine gemeinsame Software zwischen verschiedenen Versicherungshäuser quelloffen schreiben kann. Ich habe es wahrgenommen, dass es um ein Pakt 2.0 mit einem sehr spezifischem Anwendungsfall handelt. Da ging es dann um den Transfer von Freizügigkeitsleistungen zwischen Pensionskassen in der Schweiz. Das ist etwas, das braucht fast jede Pensionskasse, es gibt ca. 400 verschiedene Pensionskassen. Und jeder Arbeitgeber muss, wenn Arbeitnehmer anstellt oder kündigt, die Gelder

in die nächste Pensionskasse transferieren. Und das ist relativ viel Aufwand. Bei 400 Kassen wo es gibt, muss man schauen wo ist die Quelle und wo das Ziel. Und Open Prevo oder das MVP das wir gemacht haben, hatte das Ziel, diesen Transfer der Freizügigkeitsleistung zu automatisieren. Und stellt damit quasi ein Zusammenschluss von Versicherungen, aber innerhalb eines Landes mit einer landesspezifischen Problemdomäne. Das Konzept der Pensionskasse gibt es nur in der Schweiz. Von daher ist es eine kleine Ausprägung beziehungsweise Idee von Open Insurance, aber leider nur in der Schweiz anwendbar.

I: Open Prevo wurde gemeinsam entwickelt. Wurde es auch veröffentlicht oder nur innerhalb von den Versicherungen veröffentlicht?

B: Wenn du draufklickst landest du direkt auf den GitHub Seiten. Das war von Anfang an Quelloffen. Meine Wahrnehmung ist, wenn man neue Projekte startet, bei sogenannten Greenfield Approach ist es viel einfacher quelloffen zu starten, als Software im Nachhinein zu veröffentlichen. Denn man schreibt und strukturiert die Software ganz anders, wenn sie für eine breitere Öffentlichkeit gedacht ist. Und das ist der Ansatz den wir dort gefahren sind. Die Software ist nicht in die Produktion gegangen. Wir hatten den Status aus 6 Wochen, in welchen wir daran gearbeitet haben. 6 Wochen mit jeweils 1 Wochen Sprints. Und danach gab es eine Demoumgebung, die man den Häusern vorstellen konnte, was da passiert ist. Wo man zeigen konnte, wie und wo werden Aus- und Eintritte gemeldet. Und diese Aus- und Eintritte miteinander verbunden automatisiert. Das man sagen kann: "Hei dein Arbeitnehmer, der vorher bei dir gearbeitet hat, arbeitet jetzt gerade bei mir, schick mir mal bitte das Geld rüber"

I: Wird das Projekt jetzt weiterverfolgt oder ist es on hold?

B: Das ist aktuell einen Proposal Status, das ist korrekt. Aktuell werden Businesspläne gerechnet und geprüft wer würde eine solche Plattform betreiben und was würde sowas kosten. Im letzten Jahr ginge es darum zu zeigen, dass wir technisch bereit wären. Wir können so Software problemlos einfach gemeinsam Open Source schreiben und auch so, dass davon viele profitieren können. Was man jetzt eben ein bisschen merkt, dass es noch nicht ganz im Business angekommen ist. Zu wissen wann sich das rechnet wie man solche Cases durchrechnet.

I: Hier (Auf dem Maturitätsmodell) sieht man ja die Unterscheidung zwischen Business und Entwickler.

B: Korrekt, du referenziert gerade wieder das Maturitätsmodell. So einfach wie es ist, aber es beinhaltet wirklich prägnant, präzise die "Pain Points" die man hat, nämlich genau den Schritt zu schaffen zwischen der dritten und vierten Stufe. Also zwischen technischem Wissen von den Engineers, wie das mit dem contributen, mit der Kollaboration etc. funktioniert und daraus wirklich eine strategisches Invest zu bekommen. Dieser Schritt braucht mehr, als ein paar engagierte Engineers.

I: Kommen wir zur nächsten Frage. Was ist die Motivation von Baloise eigentlich ihre Projekte offen zu legen?

B: Also ich glaube es gibt ganz verschiedene Motivationen je nach dem wen man fragt. Du fragt jetzt jemand der (...) vom Kern her Engineer ist und weniger vom Business ist. Ich fang gleich mit der Business Motivation an. Wirklich das was häufig genutzt wird, um das Ganze zu verkaufen. Ist Kosten zu sparen. Das ist die Motivation, ob es wirklich in allen Fällen zu Kostenersparnisse im ersten Schritt führt, ist vielleicht eine andere Diskussion. Aber das ist auf jeden Fall das motiviert. Die Idee zu sagen, lass uns doch fünf Leute einfach nur einen Fünftel Investition machen und eine gemeinsame Lösung realisieren, anstatt fünf Leute ein Invest machen und alles selbst in ihren eigenen geschlossenen Teil investieren. Also dieses Potential wird sicherlich gesehen, wie schwierig es umzusetzen ist, steht auf einem anderen Blatt. Das ist mit einer der grössten Treiber im Business bei uns, ist klar wir sind eine Versicherung, rechnen alles durch und die erste Motivation ist primär Kosten zu sparen. Es gibt von der To Do Group, weiss nicht ob du sie kennst, so Guides, die verlinken wir immer in unseren Repository. Sie stellen eine ganze Reihe von Benefits dar, die zum Beispiel Unternehmen wie Google, Amazon, Facebook oder digital Natives beim Einsatz von Open Source Software sehen. Eben ich habe am Anfang gesagt, ich bin selbst zu Basler gekommen, weil ich gemerkt habe, das scheint eine andere Art Arbeitgeber zu sein, weil ich Quelloffene Dinge gesehen habe. Wir stehen bei uns mit der HR Abteilung in Kontakt, dass wir eine Arbeitgeberattraktivität dadurch bekommen, weil wir solche Projekte haben, machen und ermöglichen. Also das sind so die

beiden primären Businessnutzen. Wenn du mich als Engineer fragst. Einerseits fühlt es sich gut an, weil ich das Gefühl habe, ich tue was Gutes (...) es zu teilen. Man lernt dabei unheimlich viel. In diesem Austausch mit dem Bestreben eine Lösung zu schaffen, das nicht einem Problem genügt, sondern dem Problem vieler genügt. Also das sind auch die Punkte, die wir auf unserer Hauptseite draufhaben. Wir wollen im Prinzip lernen und weiterentwickeln. Also einfach lernen was andere machen, können und einfach ein Stück zurückgeben. (...) Ach genau, du hast jetzt gerade die Ziele und Beschreibung von dem CSTT Projekt geöffnet. Das war mal Initial ein DINA4 Dokument wo es hiess, ok das können wir uns vorstellen es als grobes Rahmenwerk für temporäres Jobwechsel vor einem halben Jahr vorzugeben. Stammt aus dem Business und repräsentiert noch eine andere Perspektive, nicht die wie bei mir eines Engineers. Steht nicht im Widerspruch dazu.

I: Wie beurteile sie die Risiken mit der Veröffentlichung von Open Source Software?

B: Also wir haben die Risiken bislang für uns als sehr überschaubar eingestuft, weil wir eben noch keine businesskritischen Komponenten veröffentlichen. (...). Ich glaube das könnte sich ändern, sobald wir grössere strategische Projekte etablieren wollen. Da ist unsere Idee, das Ganze nicht mehr so einfach als ein Projekt auf GitHub zu realisieren, einfach nur durch uns getrieben, sondern damit wirklich zur Not for Profit Organisation zu gehen. Und aktuell wäre die strategische Idee, weil wir auch stark Java und Java EE Technologie nutzen im Haus, damit zum Beispiel zur Eclipse Foundation zu gehen.

I: Also Not for Profit als neutralen Grund zu nutzen?

B: Genau, wir stellen einfach Beratungsdienstleitung dann bereit, mit einem grossen Erfahrungsschatz. Wenn es darum geht Ökosysteme zu betreuen oder Risiken abzuwägen.

I: Würden sie dann die Beratungsdienstleistungen monetarisieren wollen oder sind das dann indirekte Einnahmen?

B: Beratung unsererseits anbieten oder wenn wir Beratung in Anspruch nehmen?

I: Aha, habe es falsch verstanden.

B: Es wäre, wenn wir Beratung in Anspruch nehmen.

I: Von der Eclipse Foundation?

B: Genau. Ich weiss jetzt von Bosch selber, dass sie diese Beratungsdienstleistung selbst auch anbieten. Bei Bosch Software Innovation, sind mittlerweile so weit, dass sie ein komplettes Open Source Team und Office haben, die das selbst als Dienstleitung im Haus anbieten. Nein dann würden wir quasi die Dienstleitung zum Beispiel der Foundation in Anspruch nehmen.

I: Welche strategische Bedeutung hat Open Source für Baloise momentan?

B: Ist eine ganz gute Frage. Und ich glaube die muss man auf verschiedene Ebene aktuell beantworten. Eben auf den Ebenen des Maturitätsmodels. Auf der Ebene Contribute und Champion also das sind die Ebenen wo ein oder mehrere Projekte strategisch durch die Basler betrieben werden. Dort hat es aktuell noch keine Business Strategie. Glaube es wird erkannt, dass es strategisch sehr grossen Nutzen haben kann, wenn man solche Dinge tut, Open Prevo ist ein Beispiel, aber strategisch noch nicht verankert. Im Bereich Use sehr wohl. Also ich glaube, das ist das was man auch an den C-Levels irgendwie nach Möglichkeit Transparent machen muss. Ob man will oder nicht, sind heutzutage Systeme manchmal Open Source Software basiert und sie ex- oder implizit einfach benutzt. Je nachdem welche Umfrage man liest und was man sich dort anschaut, findet man meistens so ein Verhältnis von 80 % Open Source Software oder Komponenten auf denen eigene Software basiert und 20% stammt von einem selbst. Das sind auch Quoten, die sich bei uns so darstellen. Das ist auch frühzeitig erkannt worden. Also bei uns gibt es dort relative eine einheitliche Strategie, relativ, denn ich kann jetzt nicht für alle Projekt oder Ländergesellschaften sprechen. Aber so im Kern was ich wahrnehme ist, dass wir so wie viele andere Unternehmen auch bislang Red Hat Strategie gefahren haben.

I: Beim Use?

B: Beim Use ganz genau. Das heisst eigentlich alle Komponenten als Produkte von Open Source Projekten, das ist auch die Terminologie was Red Hat benutzt, beziehen. Und damit strategisch eine Stossrichtung haben oder hatten. Ich bin jetzt mal gespannt wie es

weiter geht, seitdem Red Hat übernommen wurde von IBM, aber genau das ist aktuell die strategische Ausrichtung. In vielen Komponenten setzten wir eben Red Hat Produkte ein.

I: Habt ihr eine allgemeine Open Source Strategie?

B: Also Strategie wäre glaube ich zu weit gegriffen, dass es wirklich eine Strategie wäre. Wir haben jetzt im letzten halben Jahr begonnen, das was bislang implizit gelebt wurde mal explizit aufzuschreiben. Das ist auch das, was in unseren Open Source Guidelines wiederzufinden ist. Diese Guidelines müssen sich erstmals bewähren. Das haben wir mal so festgehalten. Das ist was wir als Gruppe wahrgenommen haben, wie wir bislang im Kern Software genutzt oder freigegeben haben. Dass das Ganze eine strategische Manifestation hat, würde ich nicht sagen. Ich hatte das angeregt, aber es ist zuerst einmal zurückgestellt worden.

I: Aus welchem Grund?

B: Einfach, weil wir als Versicherung zu früh in diesem Prozess stecken, als dass man das schon jetzt direkt strategisch verankern möchte.

I: Ihr habt Richtlinien angesprochen. Haben sie eine Open Source Compliance und Richtlinie etabliert?

B: Wir sind zurzeit in der Etablierung. Wir haben eben jetzt mit der Forschungsstelle digitale Nachhaltigkeit oder verschiedenen anderen Ressourcen wie zum Beispiel der Eclipse Foundation Kontakt aufgenommen. Wir haben auch angeschaut wie Zalando es lebt. Das soll ein wenige herauskondensieren wie wir bislang Open Source Software genutzt und freigegeben haben. Aktuell würde ich sagen, es hat einen Status einer finalisierten Guideline, die jetzt einfach mal reflektiert angewendet werden müssen, wo man schauen muss, ob sie zum Unternehmen passen und einen Mehrwert darstellen.

I: Und habt ihr auch einen offiziellen Prozess für die Entwicklung, Veröffentlichung und Wartung von Open Source Projekten?

B: Ja im Prinzip steht das dort drin.

I: Welche Prozessstufen gibt es da so?

B: (...) Also diese Prozessstufen sind dann relevant, wenn wir eigene Projekte hosten wollen. Also es gibt diese Stufen eben. Use und Contribute, also an anderen Projekten teilnehmen, dort haben wir keine weitere Prozessschritte. In dem Bereich eigene Projekte freigeben ist es grob zusammengefasst so, man muss klären, dass initial dieser Mehraufwand, der mit der Freigabe einher geht, erstmals in Ordnung ist. Also man muss mit dem verschiedenen Stakeholder im Team, dem Product und Application Owner klären, wollen wir diesen Mehraufwand gehen. Denn häufig entsteht es erstmals als Close Source. Das heisst, wenn wir daraus Open Source machen wollen, muss man einmal über die Quellen darüber gehen, schauen inwiefern ist das Ganze entweder für die Basler ein strategischer Vorteil oder möchte aus irgendwelchen Gründen nicht open sourced werden. Das ist in den meisten Bereichen bei uns aktuell nicht der Fall, aber es werden vielleicht Bibliotheken genutzt, die nur innerhalb der Basler zu Verfügung stehen und so weiter. Wir haben jetzt gerade konkret ein Projekt, das heisst "This all That", wo wir in dieser Diskussion stehen. Das hat bei uns das BET Team in zwei bis drei Wochen umgesetzt. Jetzt haben wir angeregt, das wäre eine Komponente, die wir gut quelloffen gestalten könnten. Die Umbaumassnahmen die notwendig wären, belaufen sich glaube ich auf sieben Tage, knapp eineinhalb Wochen Aufwand, heisst 60 Stunden. Und aktuell ist die Frage, wollen wir das wirklich investieren. Also das ist glaube ich der erste Schritt, zu schätzen wie teuer ist es das Ganze quelloffen zu gestallten. Wenn das quasi in Ordnung ist und genehmigt ist, dass man diesen initialen Mehraufwand investieren möchte. Dann regen wir an, dass man das Ganze neutralisiert vom Code her, also keine Bindung an Basler spezifische Infrastruktur mehr hat. Dann im Prinzip noch die Dokumente anreichern, die so zu Sagen als Best Practice gelten. Also manche sind natürlich rechtlich notwendig. Wir müssen das Ganze unter eine Lizenz stellen. Sollte einem beschreiben, wie man daran teilhaben kann. Man muss "ReadMe"s und eine Dokumentation haben, vielleicht eine Beispielsinstallation. Teil dieser Aufbereitung ist auch zwei sogenannte Code Owner zu definieren, Leute in Projekte die zukünftig verantwortlich und als Ansprechpartner für dieses Projekt gelten. Dann schlagen wir vor, nochmals zu unserem Open Source Team zu gehen, das sind eben Matthias Kohlmann und ich. Wir gucken dann darüber und dann veröffentlichen wir es.

I: Also ihr macht dann quasi die Kontrolle?

B: Genau, also gemeinsam mit dem Team wägen wir ab, vielleicht wenn bestimmte Dinge nicht gemacht wurden, warum wurden sie nicht gemacht oder sollten sie noch gemacht werden.

I: Also das sind jetzt diese Schritte, welche sie mal in Guidelines niedergeschrieben habt, aber ist erst noch am Anfang, wie ich es verstanden habe?

B: Genau. Das ist auch in Anlehnung wie es zum Beispiel die Eclipse Foundation oder andere Unternehmen das machen. Bekämen dann den Zustand eines Inkubationsprojektes, ist im Prinzip in Inkubation befindendes Projekt, was sich erst noch bewähren muss, ob es sich wirklich über die Zeit etabliert oder ob man es wiedereinstellt.

I: Sie haben gesagt, dass ungefähr 60 Leute im Team seid und 5- 10 mittelmässen contributen. Gibt es da auch schon Rollen im Team?

B: Also nicht wirklich. Wir haben natürlich Leute in Projekten die Codeowner sind, die sich auch kümmern, wenn Bucks reported werden oder es Feedbacks von der Community gibt. Und wir haben für uns jetzt gesagt, bei dieser Grösse würde es Sinn machen, dass sich ein bis zwei Personen so bischen Hut aufsetzten. Und primäre Ansprechpartner sind oder Events organisieren, zum Beispiel wie heute. Das sind Matthias Kohlmann und ich. Und wir haben das auch in Anlehnung, wie Zalando das macht, wäre ganz gut, wenn einer von uns den Lead hätte, ist aktuell Matthias. Und ich kümmere mich so um all die Dinge, die im Community Bereich anliegen. Schaue, dass Sachen verantwortet werden, an Events vertreten sind oder erzählen was wir machen. Aber darüber hinaus haben wir jetzt noch kein Gremium, das zum Beispiel irgendwelche Freigaben macht.

I: Und arbeitet jemand bei der Baloise zu 100% nur in Open Source Projekte?

B: Nein. Nicht dauerhaft. Wir haben solche Phasen wo für paar Wochen Open Source Projekte gemacht werden, wie das MVP.

I: Ist es in Zukunft geplant ein ganzes Open Source Team zu haben?

B: Also da bin ich ein bisschen der falsche Ansprechpartner. Wenn du mich fragst, ja natürlich sollten wir das planen. Auch so ein Leuchtturmteam zu haben. Aber ich glaube aktuell konkret nein.

I: Du hast vorher noch von Codeowners gesprochen. Wenn man Open Source Software, kommen ja auch externe Anfragen, wie zum Beispiel Lizenzfragen oder sonstige Fragen. Wie gehen sie mit solchen Anfragen um?

B: Also erstmals versuchen wir schon eine ganze Reihe von Dinge von vornherein zu beantworten. Wir haben ein mittlerweile ein Template Repository das man nutzen und klonen kann. In dem sind schon Lizenzen, die wir vorschlagen, Contributionguides und "ReadMe"s drin. Dass man von der Infrastruktur her die Dinge hat, die man heutzutage erwarten würde von einem Open Source Projekt. Ich hoffe, dass so etwas wie Lizenzen oder so etwas bei uns nicht angefragt wird, wie "steht das jetzt unter dieser Lizenz oder welcher". Ansonsten nehme ich an, sie meinen jetzt sowas wie Pullrequest, Issues oder Features, die eingestellt werden. Also für gewöhnlich reagieren wir dann, zwischen ein und drei Tagen, dass wir darauf antworten. Häufig sind es Bucks oder so etwas. Und dann warten meistens bis zur nächsten Goldcard ab und setzten es dann um. Ich bin mir nicht ganz sicher, ob es die Frage beantwortet.

I: Ja, wenn sie verschiedene Codeowner habt, haben diese einen Prozess, auf was sie bei der Beantwortung beachten müssen?

B: Nein.

I: Oder macht das jeder für sich?

B: Das macht im Prinzip jeder für sich. (...) Wir haben einen zentralen Kanal mit technischen Usern, die auf jedem Projekt als Watcher drauf sind. Das heisst, wir kriegen über einen Kanal mit, was sich in allen Projekten tut. Und daher kann es relativ schnell auffallen, wenn wir uns da uneinheitlich oder doppelt Arbeit machen würden. Mir wäre nicht aufgefallen, dass es bislang passiert.

I: Du oder Matthias haben quasi die Steuerung über alle Projekte?

B: Also jeder der will kann diese Projekte watchen. Wir haben halt ein technischer Nutzer, der es bei uns in die Gruppenmailbox reinmeldet und da haben mehrere Leute Zugriff. Ansonsten kann das auch jeder im Prinzip der eine Repository watchen, da einen Überblick beschaffen. Ich habe nicht das

Gefühl, dass wir da irgendetwas aktiv überwachen müssen.

I: Habt ihr auch einen internen Kommunikationskanal oder läuft alles über GitHub?

B: Es gibt als internen Kommunikationskanal eben diese Mailbox. Meine Wahrnehmung, wieder geprägt durch die Zeit bei der Foundation. Das eigentlich im Zuge von Open Source Projekten ganz wichtig ist, offen und transparent zu kommunizieren. Und dazu gehören eigentlich auch sowas wie Mailing Lists. Wir haben uns anfangs überlegt offene Mailinglist zu verwenden, auch für solche Sachen. Die Frage kurz zu beantworten, ja es gibt interne Kommunikationskanäle, die nicht offen sind. Leider.

I: Extern habt ihr eine Mailinglist?

B: Extern gibt es leider keine Mailinglist. Da haben wir Gitter integriert, wird aber nicht genutzt. Gitter ist so ein Chatdienst, der von der Open Source Community häufiger mal genutzt wird und leichtgewichtig einzubinden ist. Kleine Snippet für GitHub Pages Projekt zum Beispiel.

I: Und wieso wird dieser nicht genutzt?

B: Das ist eine gute Frage. Ich glaube, weil die meisten, wenn sie kommunizieren sehr zielgerichtet innerhalb der Repository kommunizieren wollen. Das ist ein Kanal der übergreifend, fast schon wie eine Form von Eskalation. Wenn man sich an eine grosse Mailinglist wendet, dann kriegt man viel Feedbacks. Ich weiss nicht was einer der Gründe ist. Defacto wird es nicht viel genutzt. Ich bin da noch nicht gross nachgegangen, uns entsteht dadurch auch keine Kosten. Die Leute könnten es nutzen, wenn sie es wollten.

I: Sehen sie Bedarf die aktuelle Open Source Software zu optimieren? Falls ja, wo?

B: Ja. Also ich habe es vorher schon kurz angesprochen. Im Prinzip die Leute mitzunehmen und zu erklären welche Vorteile es hat. Und auch in diesen Modus reinzukommen, dass man versteht, hei auch ausserhalb des Baslers Universums tut sich so viel, von dem man so selber profitieren, lernen und beitragen kann. Ich glaube da ist die Versicherungswelt noch eine sehr geschützte Domäne und das wird so nicht gesehen oder erkannt. Ja ich denke, da könnte sich noch einiges tun.

I: Wie werden die Mitarbeiter im Umgang geschult? Sie haben gesagt, dass ihr die Präsentationen veröffentlicht. Aber gibt es offizielle Trainings?

B: Es gibt keine offiziellen Trainings. (...) Im Prinzip ist es immer ein aktuelles Angebot, was die Mitarbeiter wahrnehmen können. Wenn sie das Angebot wahrnehmen und Interesse haben, können sie sich melden und dann nehmen wir uns die Zeit. Und erklären es Gruppen oder auch Einzelpersonen. Wir haben alle sechs Wochen interne Fortbildungstage sogenannte Open X Days. Da gibt es immer ein Stunden Slots, um zwischen Themen Dinge zu erzählen. Und die nutzen wir häufig, um solche Themen zu besprechen und weiterzutreiben.

I: Wie kann man das Lizenzrecht den Entwicklern spannend erklären?

B: Ich glaube, man muss es stark vereinfachen. So wie wir es bei uns in unseren Guidelines auch gemacht haben. So dass man vielleicht in drei bis vier groben Kategorien denkt. Erklärt wie diese Kategorien zusammenhängen, aufeinander aufbauen, sich gegenseitig nutzen oder auschliessen. Und dann einfach erklären, dass eine Lizenz wichtig ist. Ich glaube, dass ist so das, was wir primär versuchen, anfangs zu vermitteln. Wenn es dann wirklich im Detail in spezifischen Projekten um Lizenzen ginge, dann würden wir uns sicherlich auch von Anfang an sehr intensiv mit anderen Leuten austauschen. Ich glaube das würden wir nicht alles selbst entscheiden. Wir haben mit der Rechtsabteilung schon eine Reihe von Lizenzen prüfen lassen und eine Reihe von Standard Lizenzen festgelegt, die für das daily business taugen, um zum Beispiel kleine Snippets oder kleinere Projekte freizugeben. Und bei grösseren Projekten, glaube ich, muss man dann wirklich seriös diskutieren.

I: Und welche Lizenzen braucht ihr jetzt meistens?

B: Das ist sehr unterschiedlich. Also wir fahren die Strategie, dass wir einerseits die Lizenzen des Ökösystem, in dem wir etwas veröffentlichen, uns anschauen und versuchen uns daran zu richten. Sprich in manchen Ökösystemen wird stark auf Copyleft Lizenzen geachtet. Wenn man jetzt zum Beispiel über die Eclipse Foundation spricht, werden dort oft weak Copyleft, eben die EPL standardmässig empfohlen. Es gibt aber eben auch andere Lizenzen. Im Kern ist die Strategie, dass wir

uns das Ökösytem anschauen, für das das Ganze gestaltet werden soll.

I: Was meinen sie mit Ökösystem?

B: Also zum Beispiel wie Atlassian und Confluence. Wir schauen uns an, was nutzen andere Plug-in Provider für Lizenzen, warum tun sie das und richten uns vielleicht danach. Ökösystem Eclipse Foundation würden wir jetzt sicherlich nicht versuchen MIT oder ähnliches zu forcieren, sondern würden auch sagen: Ok an den Stellen EPL, Apache oder in Teilen vielleicht sogar GPL als secondary Licences zu nutzen. Also das ist das Wort "respect to the ecosystem", indem wir uns bewegen. Wenn es in dem Fall kein Ökösystem gibt, dann fahren wir eigentlich die Strategie, permissive Lizenzen zu verwenden, die schon lange Zeit akzeptiert sind. Gotteswillen, keine eigene Lizenz zu schreiben. Dort empfehlen wir Apache 2. Das ist im Prinzip eine Standardlizenz. Zalando benutzt MIT. Der Grund weshalb wir Apache nehmen ist folgender. In Europa ist es glaube ich nicht so das Ding mit der Patentierung von Software, aber in den USA etwas stärker. Und dass man patentrechtlich zumindest ein Statement in Lizenzen auch mit drin hat, nutzen wir Apache und nicht MIT.

I: Was meinen sie mit Patenten, dass niemand anderes kein Closed Source daraus macht?

B: Also ich bin jetzt auch der Lizenzexperte. Apache sagt meiner Meinung nach explizit, dass wenn jemand ein Softwarestück mitcontributet, also angenommen Projekten von uns bekommen Contributions. Und diese Contributions sind zum Beispiel in den USA durch Patente geschützt. Dann ist es bei Apache so, dass durch dieses Contribute man implizit ein Licencegrant auf das Patent bekommt. Also man nicht sagen kann, "hei ihr verteilt hier ein Stück unserer Software und das ist in den USA patentiert, also bezahlt in den USA Patentkosten oder Lizenzgebühren für das Patent. Das wir dann dieses Patent beachten müssen. In dem Moment wo contributet wird, spricht derjenige der contributet implizit die Nutzung diesen patentrechtlich geschützen Stück Code freigibt, ohne dass man die Lizenzkosten für das Patent bezahlen muss.

I: Apache schliesst somit eigentlich das Patent aus?

B: Nicht unbedingt ausschliessen, aber man kriegt eben gleich das Nutzungsrecht dazu. Das ist meine Wahrnehmung. Apache ist ein guter, sehr guter Mittelpunkt. Von man kann gefahrlos konsumiert werden, weil man kann es modifizieren und muss es nicht wieder freigeben, aber man kann es auch konsumieren in einem Kontext wie bei der Eclipse Foundation. Die Eclipse Foundation empfiehlt natürlich primär Sachen unter EPL zu stellen, aber relativ problemlos Apache. Und viele neue Projekte nutzen auch Apache Lizenzen, anstelle der EPL. Auch um mehr Nutzung zu ermöglichen, auch wenn dann eine Closemodifkation möglich wäre.

I: Hatten sie schon Erfahrung mit Lizenzkonflikten gemacht?

B: Nein bislang noch nicht. Wir haben Tools die automatisiert mitunter auch Lizenzen bzw. die gesamte Lizenzen Zusammenstellung eines Softwareprodukts analysieren können und dort kann man mitunter erkennen, dass es sowas wie Konflikte geben könnte. Im Sinne von wenn wir zum Beispiel Software ausliefern oder verteilen würden. Da wir primär beim Konsumieren nicht wiederum verteilen, haben wir diese Konflikte nicht wirklich.

I: Welche Tools verwendet ihr? Sind es Scanning Tools?

B: Ja genau sind Scanning Tools.

I: Welche braucht ihr da?

B: Aktuell sind es noch proprietäre Tools, Nexus Lifecycle von Sonatype nutzen wir dafür zum Beispiel.

I: Kommen wir noch zum letzten Teil. Wir haben momentan über Strategie, Motivation, interne Organisation und Lizenzauswahl gesprochen. Open Source sind ja Netzwerke und Partner sehr wichtig, weil es ja die Welt öffnen sollte. Wie motiviert ihr externe bei euren Projekten beizutragen?

B: Ich glaube das ist die hohe Kunst von echtem Communitybuildung. (...) Wie wir versuchen zu motivieren ist natürlich in Marketplaces reinzukommen. Also erstmals für andere die Konsumhaltung einnehmen zu können, es so einfach wie möglich zu gestalten. Wenn wir etwas contributetn oder freigeben, dann glaube ich können andere erstmals nur mitmachen, wenn sie wiederum das nutzen, den Mehrwert erkennen und sagen "hei wir

steuern was zurück". Zurücksteuern heisst sowas wie erstmals am Anfang Bug melden. Ich glaube man stellt sich immer vor, die fixen sofort die Bugs für einen selbst und man maintain das alles gemeinsam. Das ist glaube ich in den Projekten wo wir zurzeit sind. Das passiert ab und zu, also wir kriegen sowas wie Übersetzung, Buckreports oder einen Fix und solche Dinge. Aber das ist von der Grössenordnung keine strategischen Joint Ventures die da passieren.

I: Und von wem erhaltet ihr die Feedbacks? Private Entwickler, Partner oder Konkurrenz?

B: Auch eine gute Frage. Also für uns stellt es sich das häufig so dar, als wären das Privatleute. Gefühlt sind das aber Leute, welche es in einem Unternehmen einsetzten. Wo in ihren Unternehmen nicht ganz klar ist, dürfen sie das zurück contributen. Sie contributen es dann zurück unter ihrem privaten Account und nicht als Unternehmen. Das ist das was ich wahrnehme. Aber es passiert jetzt auch nicht so häufig. Ich bin erst gerade dabei Statistik zu ermitteln, woher kommen Contribution, wann kommt Contribution und zu welchen Projekten und so weiter. Ja und wir haben dort auch keine klare zum Beispiel Contributor Licence Agreements. Also ein Vorschlag wäre, dass wir eigentlich so etwas etablieren sollten wenn es zunimmt. Das wir das Ganze auch ein bisschen steuern und uns auch selber schützen können. Das diese Contribution sauber sind. Sauber in der Hinsicht auf IP, also Intellectual Property. Meistens hält der Arbeitgeber der Person, die dort etwas zurückmeldet, dann auch das geistige Eigentum. Oder sagen wir mal die Nutzungsrechte an dem geistigen Eigentum, was er vielleicht dort freigibt. Da einfach auf der sicheren Seite zu sein. Natürlich schreckt man mit solchen Dingen wiederum Contributor ab, wenn man zuerst einmal einen Contributor Licence Agreement mit der Legalabteilung oder seinen Vorgesetzten involvieren muss. Und es ist hinderlich für Communitybuilding oder um Contributions zu bekommen. Ich glaube die Strategie dort sollte sein, sich Community anzuschliessen, in denen die Leute schon das entsprechende Mindset haben und "einfach zurückgeben". Also zum Beispiel im Kontext der Eclipse Foundation oder der Linux Foundation, wenn dort Projekte sind und da andere Contributoren sind, dann hat man eben diesen common Ground, auf dem es viel leichter fällt in der Community etwas zurückzugeben.

I: So wie ich es verstanden habe, arbeitet ihr momentan nicht schon mit einem Partner zusammen?

B: Nein. Also nicht aktiv in Open Source Projekten. Und nicht dauerhaft, wir kriegen zwar Contributions.

I: Arbeitet ihr mit einer Community zusammen?

B: Jein. Das würde ich eher verneinen. Wir sind aktiv in der Java User Group in der Schweiz, in CH Open und in der Eclipse Foundation. Das sind alles so Anfängen würde ich mal sagen, wo sich Sachen ergeben könnten. Dass es schon in echter der Zusammenarbeit darstellt, würde ich nicht unterschreiben.

I: Also seit ihr in der Annährungsphase aber noch nicht in der Zusammenarbeit?

B: Genau.

I: Würdet ihr dann eine eigene Community aufbauen oder einer bestehenden beitreten?

B: Also fragst du mich?

I: Ja

B: Denn das Thema haben wir noch nicht diskutiert. Ich würde mich sicherlich einer bestehenden anschliessen.

I: Und was ist bei einem Beitritt zu beachten?

B: Ich glaube ähnliche Gedanken, die auch wichtig sind, wenn man auf anderen Dingen beitritt oder darauf basiert. Die Maturität zu beurteilen, auf welchem Zustand befindet sich diese Community, wie funktioniert die Community und was ermöglicht die Community. Und wir haben schon viel über die Eclipse Foudation gesprochen. Das ist auch etwas was ich federführend in unserem Unternehmen vertrete und gepusht habe, dass wir diesen Schritt gehen.

I: Und warum die Eclipse Foundation?

B: Genau das wollte ich noch kurz erklären. Ich glaube viele Communities sind nicht explizit designed, um businessfreundlich zu sein. Oder für genau diesen Kompromiss von open und closed Design zu sein. Eclipse Foundation macht das schon immer so. Hat sich von Anfang an auf die Fahne geschrieben, genau diese doppelte Strategie zu unterstützen, eben

mit der weak Copyleft. Einen offenen Kern zu unterstützen trotzdem noch proprietäre Software oder Komponente dazu zu erlauben, die man dazubauen kann oder Services, die man dazu anbieten kann, etc. Ich weiss, dass geht bei anderen Lizenzen auch problemlos. Aber eben Softwarekomponenten dazuzuschreiben, die nicht quelloffen sind, das gehört zum Kernbusiness der Foundation. Ich glaube eine solche Mischung würde auch für uns zutreffen. Es ist glaube ich nicht denkbar, dass wir ein komplettes offene Unternehmen werden, aber zu grösseren Teilen mit den Tendenzen. Das deckt sich wieder an die 80/20 Regel. Einen bestimmten signifikanten Teil Quelloffen zu gestalten und dennoch 20,30,40,50 60, 70% weiterhin nicht quelloffen zu halten. Und diesen Weg zu gehen von Teilen zu öffnen, darauf zu basieren und das andere weiterhin geschlossen für sich zu halten.

I: Ich habe hier noch eine Frage zu aktivem Community Management. Aber das können wir eventuell überspringen oder betreibt ihr eine aktive Community Management?

B: Nein wir betreiben keine aktive Community. Keine eigene Community.

I: Welche Projekte werden veröffentlicht und wer entscheidet schlussendlich, ob man das darf bei euch?

B: Gute Frage. Also aktuell werden kleine Projekte veröffentlicht. Das kann jeder für sich entscheiden, fast oder man spricht mit seinem Linienvorgesetzten. Aber im Prinzip gibt es dort keine grössere Hürden zurzeit, wo Entscheide getroffen werden müssten. Ich glaube zukünftig, wenn es mehr darum geht strategische Projekte zu haben, dann wir es auf Application Owner Ebene, Product Owner Ebene, auf Linieebene oder eine Ebene darüber entschieden. Das wirklich strategisch zu verankern, Open Prevo ist ein gutes Beispiel.

I: Welcher Part vom Code wird freigegeben? Kann das momentan auch jeder selber entscheiden?

B: Die Tendenzen sind eigentlich der gesamte Code, so dass er auch Mehrwert stiftet. Ohne die Basler spezifischen Konfigurationsparameter oder Infrastrukturkomponenten von dem das Ganze abhängen würde. Das kann dann aber auch jeder selbst entscheiden. Aber bei vielen Projekten aktuell, wir sprechen manchmal davon, dass sie "verbaselt" sind. Haben sie sehr starke Abhängigkeiten die Basler Infrastruktur und können von daher nicht open sourced werden. Und dann wiederum an den Komponenten, die diese Abhängigkeiten nicht haben, denen sieht man sehr schnell an, dass man sie loslösen könnte und dann können die Teams das so selber entscheiden, dass sie das machen wollen. Wobei es manchmal meistens noch den Anstoss von aussen braucht, dass dieser Schritt auch gegangen wird. Weil es im ersten Moment einen Mehraufwand darstellt.

I: Noch zwei Fragen habe ich. Wann ist ihrer Meinung nach, der beste Moment für die Veröffentlichung?

B: Der beste Moment im Project Lifecycle?

I: Genau.

B: Meiner Meinung nach schon bei der Planung schon bei der Ideenfindung. Weil man dann, das Team und alle, Schritt für Schritt quasi damit aufwachsen können, dass das Ganze ein quelloffenes Projekt ist. Ansonsten ändert sich das Regelwerk abrupt ab einen bestimmten Punkt vielleicht. Wo werden Requirements erfasst, wie werden Issues erfasst, wie reagiert man darauf, wo wird gebaut und wo wird veröffentlicht? Also meiner Meinung nach ist der beste Punkt, bevor man vielleicht schon Requirements erfasst. Je später man es macht, desto mehr Arbeit ist zu tun, um transparent alle Dinge zu veröffentlichen. Meiner Meinung nach ist, je früher desto besser. Konkrete Erfahrung eben. Ich war 2010 ein halbes Jahr damit beschäftigt Closed Source in ein Open Source zu transformieren. Da waren wir ein halbes Jahr lang drei bis vier Leuten Fulltime beschäftigt das zu tun. Codebasis waren so zwischen 5 und 7 Millionen Zeilen Codes.

I: Und jetzt noch der Umgang mit Trittbrettfahrer oder Imitationsgefahr. Wie geht ihr damit um? Und wie differenziert ihr euch trotz Open Source vom Wettbewerb?

B: Ich sag mal diese Differenzierung trotz Wettbewerb würde ich mal zurückstellen, denn das brauchen wir noch nicht. Wir haben noch keine Sachen Open Source die direkte Konkurrenz im Wettbewerb darstellen würde, glaube ich. Es gibt die verschiedensten Formen von Trittbrettfahrern, die man sich vorstellen kann. Da wo wir das schon wahrgenommen haben und vielleicht negativ wahrgenommen haben, ist zum Beispiel bei unseren Workshops. Die werden wirklich verkauft. Die werden eins zu eins so gehalten, wie wir sie ausgearbeitet haben und werden dann einfach in Unternehmen verkauft

I: Wie verkauft?

B: Also die Dienstleistung einfach. Drei Tage Schulung von jemand anderem, von einem anderen Unternehmen in einem anderen Unternehmen. Da gehen wir eigentlich relativ gelassen damit um. Es ist schön zu sehen, dass es passiert. Es sind dann Forks von unseren Projekten. In den Forks passieren auch Fixes, Weiterentwicklung, die wir mitunter bei uns adaptieren können. Inwiefern das wiederum passiert weiss ich nicht, aber wir haben bislang keine massiven Trittbrettfahrer. Was wir nicht beurteilen können ist, inwiefern sie unsere Lizenzrechtliche Bitte nach Attribution nachkommen. Also inwiefern sie wirklich als Disclaimer oder im Nachgang sagen, Grosserteil dieser Schulung stammen eigentlich von der Basler, vielen Dank dafür.

I: Das ist ja eigentlich ein ungeschriebenes Gesetz?

B: Genau. Das kann ich nicht ganz beurteilen, wie stark das passiert.

I: Wie habt ihr erfahren, dass das passiert?

B: Also man sieht die Forks.

I: Ihr habt Folien hochgeladen und dann können sie diese downloaden, oder?

B: Ja wir haben nicht nur die Folien, sondern auch die Quellen dazu. Und die Quellen sind geforkt worden auf GitHub. In dem Fall sind es dann Kollegen oder Ex-Kollegen und die sind einfach in den Dialog gekommen nach dem Fork, dann ist es quasi so herausgekommen, dass sie sagen "ja hei wir machen die Schulung, die ist super. Wir führen sie auch durch beim Kunden". Ansonsten ist es ja immer sehr schwer festzustellen, wenn Open Source konsumiert wird, dass konsumiert wird. Man kann sowas wie Downloadstatistiken führen etc. Man kann ja nicht in die Köpfe von den Leuten gucken und wissen was sie in jeweiligen Unternehmen damit machen. Also ist sehr schwer. In dem Fall waren es persönliche Beziehungen wie wir es einfach erfahren hat, dass es passiert.

I: Ich habe auf GitHub folgende Projekte gefunden.

B: Ja wir sponsern auch eine Reihe von Projekten. Es sind so verschiedene Module hier. Das ist so die erste Stufe, dass wir zu anderen Projekten oder Dinge contributen, wie bei Jerakia, Puppet oder Liima. Also hier sind Puppet Module, die wir geschrieben haben oder eigene Komponenten, wofür wir andere Unternehmen bezahlt haben, das zu open sourcen. Das sind diese sponsored Projects.

I: Also ihr bezahlt andere Open Source Unternehmen?

B: In dem Fall von den Puppet Modulen, haben wir externe Dienstleister bezahlt. Ein Modul für Puppet zu schreiben und es open source zu gestalten.

I: Also ihr bezahlt externe Entwickler, aber nicht bei euren Projekten mitzumachen, sondern bei einer Community? Ist es eine Community?

B: Genau. Also zum Beispiel ein solches Community Modul oder eigene kleine Projekte wie Jerakia oder Liima zu schreiben.

I: Was ist der Anreiz, dass sie es via diese Communities machen?

B: Also, weil wir hier eben schon wissen, dass viele andere Unternehmen in der Schweiz, diese Software Komponente einsetzten und nutzen können. Ich kann dir über diese Projekte leider nicht so viel sagen, weil sie schon älter sind, ungefähr 2 Jahre alt. Aber das ist so etwas, was man durch relativ geringes Invest, nur monetäres Invest, ohne Zeit für Mitarbeiterausbildung etc., ein Sponsoring von Open Source Projekten in Communities hinbekommt.

I: Ihr sponsert und was kriegt ihr im Gegenzug?

B: Wir kriegen erstmals das fertige Software Produkt oder Projekt, was wir einsetzten wollen. In dem Fall Module oder Komponente, welche wir in der Infrastruktur betreiben.

I: Das wird dann wieder veröffentlicht?

B: Ja genau. Das sind 100 % Open Source Projekte.

I: Ah ok. Ihr habt das Interesse zum Beispiel für ein Plug-in und dann beauftragt ihr Externe?

B: Korrekt.

I: Und die machen das im Rahmen von dieser Community?

B: Schon direkt quelloffen, genau.

I: Ihr stellt einfach Geld zur Verfügung?

B: Ja genau. Wir bezahlen einfach die Externen. Die machen wiederum Attribution und sagen "Vielen Dank, gesponsert wurde das Ganze durch die Basler".

I: Wie wird das von der Community dann aufgenommen, wenn etwas gesponsert wird?

B: Gefühlt sehr gut. Zum Beispiel dieses Puppet Modul, da sind wir letztens durch Herr Henkel aufmerksam gemacht worden, dass es die Referenzimplementierung von Red Hat für Firewall Module, Firewall Management über Puppet ist. Und hat 350 000 Installationen weltweit. Wird unter anderem im CERN eingesetzt. Ich glaube das wird schon sehr gut angenommen.

Interview mit Henning Henkel und Tobias Denzler am 17.01.2019

I: Zuerst ein paar Worte zu mir. Ich bin Masterstudent und stehe jetzt vor dem Abschluss. Und für das muss ich eine Forschungsarbeit machen. Das Thema ist: Open Source und neuen Players in der Open Source Bewegungen, also Unternehmen oder Behörden, welche ihre Tätigkeit nicht primär in der Softwareentwicklung haben oder von der IT Industrie stammen. Auf sie bin ich durch Markus Tiede gekommen, weil ich ihn gefragt habe, ob er noch andere Unternehmen kennt. Dann können wir die Vorstellungsrunde weiterfahren. Also mich würde interessieren, wer ihr seid, was ihre Ausbildung ist, wie ihr zur Helvetia gekommen seid und was eure aktuelle Funktion ist.

B: Herr Denzler: Mein Name ist Tobias Denzler. Ich habe einen Masterabschluss in Computer Science in Basel gemacht. Vor der Helvetia war ich bei der SBB. War dort im Team, welches die OpenShift und Cloud Plattform aufgebaut hat. Dort habe ich zum Teil auch schon Sachen wie Open Source Strategien innerhalb der SBB gemacht. Meine aktuelle Position bei der Helvetia ist Plattform Solution Architekt. Auch wieder für OpenShift und AWS.

I: Was ist AWS?

B: Herr Denzler: Amazon Web Service. Und da sind wir mit unserem Team im Aufbau und Betrieb dieser Cloud Plattform.

B: Herr Henkel: Mein Nami ist Henning Henkel. Ich bin Diplominformatiker. Ich habe an der FH Furtwangen in Deutschland studiert, war davor bei der Baloise, war dort zeitlang Fachteamleiter im Bereich Linux unter anderem. Meine jetzige Position hier bei der Helvetia nennt sich Automation Solution Architect. Und aktuell habe ich den Auftrag, eine Analyse oder ein Projekt zu machen, wo es um die Einführung einer zentralisierten Automatisierungsplattform geht, auf Basis von Ansible Tower.

I: Dann fange ich mal mit den Fragen an. Wie ist die Idee bei der Helvetia entstanden, eigene Projekte, Software oder Teile der Software zu veröffentlichen und der Allgemeinheit kostenlos zur Verfügung zu stellen?

B: Herr Denzler: Also wir sind eigentlich noch gar nicht so weit (lacht). Also prinzipiell setzten wir sehr viel Open Source selbst ein. Eben mit Ansible, Open Shift oder auch viel auf der Entwicklungsseite. Und dort wollen wir gewisses zurückgeben. Und in eurem Bereich (Herr Henkels Bereich) ist es aktuell so, dass es einfacher ist. Also wenn du kontributest, dann quasi das Upstream nehmen kannst und deine eigenen Versionen auch nicht noch maintainen musst, oder?

B: Herr Henkel: Genau richtig. Also das ist sicher einer der Treiber dafür. Wir nutzten recht viel Open Source Software in gewissen Bereichen und profitieren davon. Teilweise entspricht es noch nicht dem, was wir brauchen. Und dann haben wir die Wahl. Entweder ich Pflege meine eigen Branch Inhouse und muss den über den ganzen Lifecycle pflegen oder ich schaue, dass ich den Fix Upstream kriege. Dann muss ich mich nicht mehr darum kümmern, meine Modifikation selbst zu pflegen, sondern es ist in diesem Upstream Projekt schon drin und ich kann einfach davon profitieren. Andere profitieren davon auch und so gewinnen alle. Und ich denke, wir sind inzwischen vielfach an dem Punkt, dass man sich nicht mehr durch die eingesetzte Software unbedingt einen Vorteil verschafft. Also bei uns geht es um Versicherungen zu verkaufen. Ob wir nun die gleichen Ansible Module einsetzten wie die Baloise oder nicht, spielt wahrscheinlich am Ende nicht so die grosse Rolle. Und es muss niemand das Rad neu erfinden.

I: Mit Upstream meinen sie, dass sie es einer Community übergeben und es nicht selbst pflegen zu müssen, oder?

B: Herr Henkel: Genau, richtig. Ja also in dem Bereich, wo ich tätig bin, ist meine Massgabe aktuell, dass wir nach Möglichkeit nicht immer alles neu schreiben und schauen, was ist schon da und das nutzen. Und wenn es notwendig ist, eben dann auch in der Community oder bei dem Maintainer. So probieren wir unsere Weiterentwicklung oder Änderungen reinzukriegen.

I: Kommt der Wunsch zu Open Source Software beizutragen von den Mitarbeitern bzw. Entwickler oder von dem Management aus?

B: Herr Henkel: Ich denke es ist eher Bottom-up. Und das Management hat, glaube ich, noch nicht richtig eine Idee, was es bedeutet für die Firma.

B: Herr Denzler: In unserem Fall ist es so, dass das Management dagegen nicht abgeneigt ist. Aber sie warten zuerst mal ab und schauen, was bedeutet das überhaupt. Aber es ist schon mehrheitlich getrieben von Entwicklern, das auf jeden Fall.

I: Haben sie dort Schwierigkeiten das Management zu überzeugen oder auf was warten sie?

B: Herr Denzler: Nein, es ist aktuell mehr ein warten von unserer Seite, dass wir quasi zuerst die ganzen Rahmenbedingungen setzten müssen. Mal grundlegende Sachen abklären und für uns eine Idee haben, wie wir eine solche Contribution vorstellen. Was wären sinnvolle Use Cases. Welche Abklärungen mit Legal und Compliance gemacht werden müssen. Und wann man dann ein Startpaket zusammen haben, würden wir mal zum Management gehen und es vorstellen.

B: Herr Henkel: Genau. Ich denke, wir sind halt inzwischen auch an einem Punkt, dass es dem Management klar ist, dass Open Source Software eingesetzt wird. Die Frage ist dann halt immer, ob es eine Community-driven Open Source Software oder eine kommerziell supportete Open Source Software ist, die wir einsetzten, wie zum Beispiel der Firma Red Hat. Bisher ist man sehr stark auf dem Commercial Support Zug, sage ich mal. Aber in gewissen Bereichen wird man nicht um den Einsatz von Community Software kommen. Wie man damit umgeht, ist völlig unklar. Was wir dann auch zurückgeben wollen und was die Rahmenbedingungen für uns generell sind.

I: Was für Projekte, Software oder einzelne Codes habt ihr unter einer Open Source Lizenz entwickelt und veröffentlicht oder planen sie zu veröffentlichen?

B: Herr Denzler: (...). Veröffentlicht haben wir noch eigentlich nichts.

B: Herr Henkel: Genau, also nicht direkt.

B: Herr Denzler: Ja nicht direkt, genau (lacht).

I: Was wäre dann indirekt?

B: Herr Henkel: Indirekt heisst, dass du als Mitarbeiter kannst natürlich an Open Source Software mitarbeiten und contributen. Du kannst es natürlich auch während der Arbeitszeit machen, aber ist aber nicht unbedingt eine direkte Contribution durch die Firma.

I: Dürft ihr während der Arbeitszeit in anderen Communities contributen?

B: Herr Denzler: Ja das ist halt immer so ein Zwischenbereich. Also wenn es Sachen sind, wo du sowieso damit arbeitest. Dann ist es nur noch eine Frage, ob und wie du deine Arbeit, welche du sowieso machst, wieder irgendwo zurück contributest. In dem Fall macht man das halt zum Beispiel mit dem privaten Account und contributet dann als Privatperson und nicht als Helvetia Mitarbeiter.

I: So wie ich es verstehe, habt ihr momentan eine Zwischenlösung, wie ihr der Community etwas zurückgebt?

B: Herr Henkel: Das Problem ist, dass es aktuell nicht definiert ist wie man damit umgeht. Das ist auch einer der Gründe, warum wir das Thema aufgenommen haben und mit der Baloise oder mit der SIX usw. in Kontakt sind. Und da mehr oder weniger eine einheitliche Linie unter den Firmen hinzukriegen, weil die alle schlussendlich das gleiche Problem haben. Und die Baloise ist ja dann schon ein Stück weiter. Die arbeiten ja mit der Universität Bern zusammen.

I: So wie ich es verstanden habe, habt ihr noch nichts direkt veröffentlicht. Ich habe auch keinen GitHub Account von der Helvetia gefunden, vielleicht habe ich auch schlecht gesucht.

B: Herr Denzler: Nein, den gibt es noch nicht.

I: Ist er in Planung?

B: Herr Henkel: (...) Also mittelfristig wahrscheinlich schon, ja.

I: Dann gehen wir zum Thema Motivation rüber. Was würdet ihr sagen, was ist die Motivation von der Helvetia, eigene Projekte oder Software zu veröffentlichen? Ihr habt vorher schon gesagt, dass ihr etwas zurückgeben wollt. Habt ihr noch weitere Chancen, die ihr durch die Veröffentlichung seht?

B: Herr Denzler: Also aus meiner Sicht sind es zwei Hauptpunkte. Das eine ist vielleicht aktiv oder aktiver die Entwicklung von einem Produkt oder Open Source Komponente zu steuern, die man sowieso einsetzt. Man hat dann eher die Möglichkeiten gewisse Features

in ein Produkt rein zu bringen oder Bugs oder Fehler, die einen im daily business verhindern, dass man die schneller geflickt kriegt, als man wartet darauf bis es jemand anderes macht. Das zweite ist auch ein bisschen die Sicht von aussen. Also wie ist das Standing aus Sicht eines Entwicklers? Wie attraktiv ist eine Firme für einen Entwickler dort zu arbeiten? Ich denke, einige Entwickler sagen "ok eine solche Strategie ist ein Plus".

B: Herr Henkel: Also ich denke die Attraktivität des Arbeitgebers ist ein Punkt, der da positiv sein könnte. Das andere ist die generelle Wahrnehmung in der IT Industrie. Als Finanzdienstleister hat man da nicht unbedingt den besten Ruf. Dass man hauptsächlich konsumiert und nichts wirklich zurückgibt und ich denke, wenn man sich in dem Umfeld tummelt, hat es da vielleicht auch einen positiven Effekt.

I: Wie sieht es mit den Kosten aus?

B: Herr Henkel: Also das sehe ich jetzt eher als zweitrangiges Thema. Ich bin gerade daran einen Entscheid vorzubereiten. Da geht es darum, entweder wir machen Full Open Source oder wir nutzen alles von der Community. Dann brauche ich aber das Knowhow intern, das kostet mich Geld. Wenn ich nicht so viel Knowhow habe, dann will ich eher Commercial Support haben. Da unser Kernbusiness nicht unbedingt auf Softwareentwicklung liegt, denn wir sind eine Versicherung und damit verdienen wir das Geld. Versicherungen tendieren in der Regel eher dazu den kommerziellen Support zu nehmen und dafür vielleicht weniger Personalkosten zu haben.

I: Dann zahlt man lieber zum Beispiel Red Hat für die Hilfe, für das Implementieren?

B: Herr Denzler: Genau.

B: Herr Henkel: Weil schlussendlich, wenn man schaut was man für Support ausgibt und was ein Mitarbeiter kostet, kann ich relativ viel Subscribtions oder Support beziehen für das Geld was ich für einen Mitarbeiter zahlen würde. Mit einem Mitarbeiter kommen wir ja noch nirgends hin, das ist so ein bisschen das Problem. Wenn ich natürlich eine Firma bin, wo das Knowhow schon da ist, dann mag das anders aussehen. Würde zumindest in der Versicherungsindustrie sagen, ist es in der Regel nicht so weit verbreitet.

I: Welche Risiken sehen sie bei der Veröffentlichung von Open Source Software?

B: Herr Denzler: In unserem Fall am ehesten irgendwelche Legalthemen. Dass die Leute keine Ahnung haben, was es bedeutet, wenn man sich für eine gewisse Lizenz entscheidet. Klar, wenn man Softwares veröffentlicht, welche instabil ist oder Bugs haben, ist es nicht förderlich für deinen Ruf. Aber ich denke wir sind nicht in der Position, wo wir extrem viel Komponenten open sourcen können.

B: Herr Henkel: Ja ich denke da sind wir aktuell noch nicht exponiert. Wir sind keine Allianz, die sagt, dass sie das ganze Kernversicherungssystem veröffentlicht. Uns geht es mehr darum einzelne Komponente von Infrastruktur vielleicht zu contributen. Da denke ich auch, dass die Risiken relativ gering sein dürfen. Die Legalthemen natürlich.

I: Wenn ich sie richtig verstanden habe ist die Hauptmotivation, dass man Lösungen aktiver steuern kann, welche aktuell noch nicht 100% zu euren Bedürfnissen passen. Und das Image, im speziellen, um IT Spezialisten anzuziehen.

B: Herr Henkel und Herr Denzler: mhm (bejahend).

I: Welche strategische Bedeutung hat Open Source Software momentan?

B: Herr Henkel: Das ist noch nicht klar. Es gibt noch keine strategische Bedeutung für Open Source.

B: Herr Denzler: Ja, vielleicht nicht direkt. Ich meine indirekt schon. Wir haben sehr viel Linux Systeme. Gerade im Cloudbereich, wo wir ausschliesslich auf Open Source setzten, klar mit kommerziellem Support, aber im Hintergrund ist alles Open Source. Schlussendlich, im weitesten Sinn, ist jede Java Applikation mit Jabbers mittlerweile auch Open Source.

B: Herr Henkel: Also ich denke, dass ist zwar Fakt, dass die Tools eingesetzt werden, aber ich glaube nicht, dass sich die Firmen entschieden hat, sie zu verwenden, weil sie Open Source sind.

B: Herr Denzler: Ja.

I: Was war denn der Grund, dass man jetzt Linux verwendet?

B: Herr Denzler: Ja, das ist mehr eine Frage der Möglichkeiten oder der technischen Anforderungen.

B: Herr Henkel: Ja, ich denke, wenn man in der Historie schaut bei Versicherungen, waren die in der Regel auf kommerzielle Unix Derivaten wie Solaris, HPUX oder sonstiges. Und die Produkte sind in der Regel tot. Entweder zahlst du sehr viel Geld. Und einfach da hat Linux gewonnen. Der Logische Nachfolger von diesen Systemen war Linux, weil sie günstiger sind und inzwischen weiterverbreitet sind. Ich sage mal, der normale Informatikstudent, der sich irgendwie mit einem solchem System auseinandersetzt, kennt eher Linux System als HPUX oder Solaris. Und es zieht sich halt durch. Ich denke eben, es gibt inzwischen eine gewisse Marktmacht. Dass Firmen halt auf die Produkte setzen, weil die bisher führenden Anbieter nicht unbedingt die besseren Produkte haben. Wenn man zum Beispiel Jabber mit Weblogic vergleicht, dann ist Jabber sicher günstiger wie Weblogic. Aber Jabber hat über die Jahre auch an Möglichkeiten aufgeholt.

I: Ihr habt vorher erwähnt, dass ihr noch am Ausrechnen seid, ob man mit Community oder mit kommerziellen Anbietern wie Red Hat arbeiten soll. Spielt die Abhängigkeit mit einem kommerziellen Anbieter wie Red Hat auch eine Rolle? Also wenn ihr selbst mit einer Community zusammenarbeitet, dass ihr weniger abhängig von Drittdienstleister wärt.

B: Herr Henkel: Nein. Aus meiner Sicht ist in dem Bereich, wo ich es anschaue, kein Thema. Weil zum einen ist es Open Source und da habe ich immer eine gewisse Abhängigkeit zu Red Hat, jetzt in diesem Fall, weil sie treiben das Produkt. Es ist mehr die Frage, wenn ich ein Problem habe, ob ich selbst das Knowhow Inhouse habe. Will ich dies aufbauen, will ich das bezahlen, will ich eine Fallbackebene haben bei Red Hat oder bei jemand anderem. Dadurch, dass es Open Source ist, kann ich zu Firma XY gehen, um das Produkt zu warten. Ob das jetzt sinnvoll ist, sei mal dahingestellt. Die Abhängigkeit an sich ist da weniger ein Thema. Die Frage ist aus meiner Sicht, für was man das Geld ausgeben. Will ich das Geld für Software ausgeben und mich auf meine Kernkompetenz besinnen oder habe ich das Gefühl, dass ich das IT Knowhow aufbauen will, dann gebe ich das Geld dort aus. Da bei uns aber strategisch eher in die Richtung gedacht wird, dass man die sogenannte Fertigungstiefe verringern möchte, gehe ich davon aus, dass das Management eher für Fullsupport von Red Hat votieren würde.

I: Habt ihr bei Red Hat ein Subscribtionsangebot? Also ihr zahlt monatlich wie bei einem Abo, dafür erhaltet ihr Support?

B: Herr Henkel: Jährlich genau.

I: Und es ist jährlich kündbar?

B: Herr Henkel: Genau. Red Hat hat praktisch nur dieses eine Model.

I: So wie ich es verstanden habe, existiert keine offizielle Open Source Strategie?

B: Herr Henkel und Herr Denzler: Ja.

I: Und es ist auch nicht in einer IT- oder Firmenstrategie integriert?

B: Herr Henkel: Bisher nicht nein.

I: Kommen wir zum Thema über die interne Organisation. Wie viele Leute bei der Helvetia beschäftigen sich mit Open Source Software?

B: Herr Denzler: Also es kommt ein bisschen darauf an.

B: Herr Henkel: Was man unter beschäftigen versteht. Wir haben natürlich Leute, die Linux Systeme aufbauen, aber die contributen nicht direkt an eine Community. Sondern sie bauen Linux Systeme auf und dann gibt es Leute, die bauen Windows Systeme auf.

B: Herr Denzler: Also ich denke, jeder IT Mitarbeiter verwendet und betreibt Open Source Software.

I: Wie viele IT Mitarbeiter habt ihr ungefähr?

B: Herr Denzler: Ungefähr 350. Aktiv zu Open Source contributen, sind es vielleicht fünf bis zehn. Ein relativ kleiner Teil.

I: Ihr habt jetzt auch kein Open Source Team oder so?

B: Herr Henkel: Nein. Wie gesagt, wir sind da ganz am Anfang. Wir sind zum Beispiel auch aus unterschiedlichen Ressorts. Dadurch dass es bei uns ein Thema ist und wir sehen, dass wir da was machen müssen, sind wir da mal

langsam daran irgendwie vorwärts zu machen, aber bisher war das kein Thema und da hat sich keiner mit auseinandergesetzt.

I: Und ihr arbeitet jetzt neben eurer Tätigkeit an diesem Thema?

B: Herr Denzler: Genau.

B: Das ist so. Es ist jetzt nicht wie zum Beispiel bei Markus, dass der gewisse Prozentsatz dafür angestellt ist. So etwas gibt es bei uns nicht.

I: Ist ein Open Source Team zukünftig geplant, welche das Thema antreibt?

B: Herr Denzler: Also ein direktes Team wahrscheinlich nicht. Ich könnte mir vorstellen, dass es eher querschnittmässig sein wird. Dass man aus verschiedenen Bereichen verschiedene Leute hat, wo sich mehr als eine interne Community das ein wenig treiben. Aber schlussendlich ist in unserem aktuellen Fall ein Team noch nicht notwendig. Wir brauchen jetzt viel mehr mal Richtlinien, klare Verhältnis, wie machen wir es, was machen wir und das Einverständnis von oben. Wenn wir das geklärt haben, kann dann auch jeder, der will, dazu beitragen.

B: Herr Henkel: Genau. Ich denke eben, es geht da viel um Richtlinien. Wie was sind Lizenzen für uns? Welche Software können wir dann open sourcen? Wie ist unser Engagement in Community und so weiter und so fort.

B: Herr Denzler: Ich denke, das würde sich ein wenig verändern, wenn man effektiv eine eigene Softwarekomponente zum open sourcen hat. Wie zum Beispiel ein Versicherungsrechner oder so etwas. Dann sieht es sicher anders aus.

I: Wie ich es verstanden habe, seid ihr gerade Compliance und Richtlinien am erarbeiten?

B: Herr Denzler: Ja, also in einer sehr frühen Phase. Wir haben erste Abklärungen mit Legal und Compliance gehabt. Und bauen es nun auf dem auf.

I: Wie würde der Prozess bei der Entwicklung, Veröffentlichung und Wartung aussehen? Habt ihr euch dazu schon Gedanken gemacht?

B: Herr Denzler: (...) Also unsere Herangehensweise ist, dass wir es so einfach wie möglich halten. Wir wollen nicht viel Prozess dahinter stellen. Wir wollen Richtlinien machen, wo die Entwickler sehen, welche Lizenzen sie nutzten dürfen und welche nicht. Und vielleicht noch, dass man keine Passwörter oder sensitive Daten veröffentlicht usw.

I: Also so einfach wie möglich, aber gleichwohl braucht es vor der Veröffentlichung gewisse Zwischenschritte, die gegangen werden müssen?

B: Herr Denzler: Du sprichst jetzt mehr die Veröffentlichung von eigenen Komponenten an?

I: Ja genau.

B: Herr Denzler: Da bräuchte es dann wahrscheinlich schon einen stärker definierten Prozess.

B: Herr Henkel: Das denke ich auch. Aber bei uns aktuell kein Thema und dementsprechend haben wir da keine Prozesse.

I: Wo sieht ihr den grössten Bedarf, um Open Source Software zu veröffentlichen zu können? Welche Hebel müssen angezogen werden, um dies zu erreichen? Was sind die nächsten Schritte?

B: Herr Henkel: Wir sind jetzt mit anderen Firmen im Austausch. Es wird einen Workshop geben, welchen die Red Hat begleitet. Und da wird man mal schauen, ob wir eine Art Richtlinien langfristig definieren.

B: Herr Denzler: Am Anfang wird es darum gehen, das ein mal eins zu definieren. Was sind die Richtlinien? Mit Legal und Compliance zu schauen, unter welchen gesetzlichen Voraussetzungen können wir das machen. Wenn man dort einen Nenner findet, dann ist aus meiner Sicht die grösste Arbeit gemacht. Danach geht es mehr darum, wie wir unsere Entwickler enabeln, dies mehr zu brauchen. Oder mit paar Beispielen oder paar Use Cases aufzeigen, dass es einen Mehrwert hat schlussendlich.

I: Mit Richtlinie habe ich eigentlich interne Richtlinie gemeint. So wie ich es verstanden habe, ist ihr Ziel eine Branchen- oder Industrierichtlinie zu etablieren? Zum Beispiel mit der SIX Group oder der Baloise zusammen?

B: Herr Henkel: Baloise entwickelt ja ihr ganzes Open Source Framework auf GitHub,

weil sie sagen, dass es keinen Sinn dies hinter verschlossener Tür zu machen. Wir sehen das ähnlich. Wir müssen das Rad nicht neu erfinden. Ich meine, wenn sie schon viel Zeit rein investiert haben. Das ist ein Stück weit die Idee von Open Source, dass man Sachen teilt und voneinander profitiert. Genau den Ansatz wollen wir halt mit den Richtlinien ein Stück weit auffahren.

I: Haben sie Richtlinien wie externe Komponente in eure interne Software oder Infrastruktur integriert werden darf?

B: Herr Denzler: Ich denke nicht, oder?

B: Herr Denzler: Ich denke es auch nicht. Bei uns liegt ja in der Regel nicht der Fokus darauf, dass es Open Source ist, sondern in der Regel gibt es bei uns ein Projekt, um ein Ziel zu erreichen. Dann kann es zufällig sein, dass es Open Source ist. In der Regel ist das ein Produkt mit kommerziellem Support. Dann gibt es ein Projekt, einen Entscheid und damit halt die Freigabe das Produkt einzusetzen.

B: Herr Denzler: Wo wir aktuell dran sind, ist mehr auf Library Ebene, dass wir dort Lifecycle oder Security Scanning machen. Dass man zum Beispiel weiss, ob alle Komponente, welche wir einsetzten, die richtige Lizenz haben. Oder gibt es eine, welche es uns nicht mehr ermöglicht. Das andere sind mehr Lifecycle Themen, wie zum Beispiel ob es sich um eine völlig veraltete Version handelt.

I: Braucht ihr dazu sogenannte Source Code Scnanning Tools dafür?

B: Herr Denzler: Ja, nicht direkt Source Code Scanning. Mehr so Lifecycle oder Security Scanning. Aktuell sind wir dies am Evaluieren und schauen uns die komplette Palette an.

I: Ihr habt gesagt, dass ihr noch nichts veröffentlicht habt. Falls ihr etwas veröffentlichen würdet, welche Lizenzen würdet ihr dazu verwenden?

B: Herr Henkel: Ich finde es schwer dies zu beantworten, denn zum einen gibt es so viele Lizenzen und zum anderen gibt es so viele Impacts, je nachdem für was man sich entscheidet. Ich denke nicht, dass wir das aktuell abschliessend fixen könnten.

I: Aber wenn man mal in den drei Clusterkategorien denkt zwischen starkem Copyleft, schwachem Copyleft und permissiven Lizenzen. Bei welchen seht ihr Vorteile und Nachteile?

B: Herr Denzler: Also in unserem Fall, weil wir nicht wirklich eigene Komponente veröffentlichen, sondern eher contributen. Dann ist es sehr wichtig zu schauen, dass die Lizenz dich nicht einschränkt. Dass man eine Lizenz wählt, wo plötzlich die Pflicht besteht den gesamten eigenen Code veröffentlichen zu müssen.

B: Herr Henkel: Prinzipiell sind Lizenzen wie die GPL spannend, weil du einen Rückfluss kriegen solltest. Und die Community am ehesten davon profitiert. Wenn man es aus kommerziellen Gesichtspunkten anschaut, wäre die GPL nicht unbedingt die beste Wahl.

I: Welche Lizenz würde beidem gerecht werden, der Community und den kommerziellen Gesichtspunkten?

B: Herr Henkel: Also dazu bin ich zu wenig in der Thematik drin, um zu sagen, dass ich lieber die Apache Lizenz statt der MIT verwenden würde.

B: Herr Denzler: Dadurch, dass Softwareentwicklung nicht unser Corebusiness ist und wir primär nicht mit Software Geld verdienen, ist dort eine Einschränkung über eine Lizenz auch nicht zwingend notwendig. Denn wir würden es nicht aus finanziellen Gründen open sourcen, sondern mehr aus ideologischen Gründen oder aus Gründen, welche unsere Arbeiten einfacher machen. Daher würde ich eher zu einer relativ offenen Lizenz tendieren.

B: Herr Henkel: Aus der Erfahrung, die ich beispielsweise bei der Baloise gemacht habe, wo wir über einen Dritten Software praktisch open sourced haben, kann ich folgendes sagen: Es war so, dass wir intern sehr lange eine niedrige Version gefahren haben. Die Community hat es aufgegriffen und sehr stark erweitert und wir haben schlussendlich davon profitiert. Ich denke, wenn die Lizenz sowas unterstützt, ist das natürlich gut.

I: War das mit der Poppet Community?

B: Herr Henkel: Genau, richtig

I: Habt ihr dort auch mitgesponsert?

B: Herr Henkel: Ja.

I: Also war das ein Sponsoring zusammen mit der Baloise?

B: Herr Henkel: Nein. Das war, als ich noch bei der Baloise war. Da gibt es verschiedene Poppetmodule. Zum Beispiel für Firewalls gibt es ein Modul, was die Baloise gesponsert hat. Was dann eben aufgegriffen wurde von der Community und inzwischen soweit ich weiss im CERN und im europäischen Parlament eingesetzt wird. Die Community hat ganz viele Features implementiert und inzwischen ist das Modul, das Einzige was von Poppet selbst recommended wird. Das war für die Firma eine win-win Situation. Man hatte das initiale Sponsoring übernommen, hat es einem Dritten gegeben, der die Pflege übernommen hat und die Community hat es aufgegriffen. Die Baloise selbst hat die Feature nicht benötigt, aber es wurde praktisch weiterentwickelt.

I: Machen sie bei Helvetia auch so Sponsoring an Communities?

B: Herr Henkel: (...) indirekt teilweise. (...) Also wir haben bisher keine Produkte, wo unser Name jetzt dran steht. Aber jetzt Beispiel bei Jerikia, da haben wir schon auch gesponsert.

I: Und wieso habt ihr gesponsert, ohne dass euer Name draufsteht? Wo ist da der Mehrwert?

B: Herr Henkel: Weil wir bisher noch keine Strategie haben, wie wir damit umgehen.

I: Und wieso sponsert ihr aber dann? Was kriegt ihr für eine Gegenleistung für das Sponsoring?

B: Herr Henkel: Also in dem konkreten Fall ging es um ein Problem, das wir gelöst haben wollten. Das Sponsoring sieht dann so aus, dass wir den Entwickler bezahlen, dass er die Features implementiert und die wandern nachher in das Open Source Produkt.

I: Ist dieses Sponsoring günstiger als wenn man einen kommerziellen Entwickler anstellen würde?

B: Herr Henkel: Weiss ich nicht, ob sich so etwas schenkt. Es ist nachher eine Frage der Verfügbarkeit von Tools. Das sind auch alles relativ spezielle Sachen, würde ich sagen.

I: Ihr geht mit dem Sponsoring Ansatz einen Kompromiss ein, weil ihr das Eigentum auch an die Community übergebt?

B: Herr Henkel: Das ist so. Da sind wir wieder bei dem Thema, muss ich das Rad immer neu erfinden? Und wo ist der Mehrwert? Ich sehe es eher so: Jetzt hat es die Community und die Pflege ist gesichert. Und wir als nicht Softwarehersteller müssen uns nicht vordergründig darum kümmern.

B: Herr Denzler: Ja genau.

I: Und habt ihr die Features gekriegt, die ihr wolltet?

B: Herr Henkel: Ja genau. Also in dem Fall waren es jetzt fünf Tage, wo der hier in Basel war und bei uns praktisch an dem Feature gearbeitet hat.

I: Spannend. (...). Hattet ihr schon Lizenzkonflikte bei der Benutzung?

B: Herr Denzler: Ja, das ist schwierig zu beantworten, weil wir das auch nicht prüfen. Könnte mir vorstellen, dass es Konflikte gibt, wenn wir es anschauen würden. Das andere ist natürlich, das ganze Thema wird immer komplexer. Wir haben bei der Helvetia eine Cloud First Strategie und das heisst, du hast weniger Kontrolle über diese Sache. Gerade bei Lizenzen wie AGPL, wo allein nur durch die Verwendung der Software eines Drittanbieters wie zum Beispiel „Software as a Service“, plötzlich Auswirkungen auf andere Komponenten haben kann. Das ist etwas, was heutzutage gar nicht abgedeckt ist.

I: Musstet ihr schon Codes zwangsveröffentlichen?

B: Herr Denzler: Nein, wüsste ich nicht.

B: Herr Henkel: Wüsste ich auch nicht.

I: Wie ich es richtig verstanden habe, gibt es keinen Prozess, wo ihr das überprüft?

B: Herr Denzler: Nein, noch nicht. Ein Scanning auf Lizenzen ist sicher angedacht, damit man die schlimmsten herausfiltern kann.

I: Kommen wir zum Thema Training. Wie werden Mitarbeiter für den Umgang mit Open Source Software geschult und sensibilisiert?

B: Herr Denzler: Gar nicht eigentlich (lacht). Ich denke, das wäre auch ein Thema für die Richtlinien. Wie gesagt, wir stehen noch ganz am Anfang. Das wäre sicher ein Punkt, welcher

noch einfliessen würde. Wie wir das Thema Open Source, als Firma, als Entwickler und als Community angehen wollen.

I: Offizielle Trainings bietet ihr somit nicht an? Aber gibt es zum Beispiel Lunch Talks oder sonst Freiwillige, welche versuchen es zu pushen?

B: Herr Henkel: Wie gesagt, wir sind da noch ganz am Anfang. Die Awarness besteht noch nicht.

I: Was denkt ihr, wie könnte man es den anderen Entwicklern spannend erklären und eben diese Awarness schaffen?

B: Herr Henkel: Dazu muss man gewisse Richtlinie und Framework haben, wo man definiert was die Dos and Don'ts sind. Es besteht die Möglichkeit von verschiedenen Formaten. Es kann eine interne Schulung sein, die eher einen offiziellen Charakter hat. Dann gibt es Überlegungen, ob man kurz- bis mittelfristig so etwas einführt wie es die Baloise zum Beispiel macht. Die nennen es Open X Days, wo die Leute dann wählen können, welche Präsentationen sie sich anschauen. Dann sind natürlich irgendwelche Lunchtalks möglich. Müssen dann schauen, welche Formate da passt.

I: Bei Open Source Software sind Partner ziemlich wichtig. Ihr habt vorhin die Baloise oder SIX Group erwähnt. Mit welchen Partnern arbeitet ihr zusammen?

B: Herr Henkel: Wir sind erst im Kontakt. Eine Zusammenarbeit ist das noch nicht. Das ist ein grosses Wort (lacht).

B: Herr Denzler: Sind noch nicht so weit (lacht).

B: Herr Henkel: Also wir sind jetzt mit der Baloise, Six, Postfinance, Mobiliar und Red Hat in Kontakt.

I: Wie habt ihr eure Partner, nenne sie mal so, ausgewählt? Nach welchen Kriterien habt ihr sie kontaktiert?

B: Herr Henkel: Durch persönliche Kontakte, denke ich. Ich komme von der Baloise und kenne Matthias Kohlmann sehr gut und so sind wir mit der Baloise zusammengekommen. Red Hat haben wir hier im Haus. Die SIX waren an der Linux Erfa, also Linux Erfahrungstag, das ist zweimal im Jahr. Und da haben sie eine Präsentation über Open Source im Unternehmen gemacht und haben darum gebeten, dass man sie kontaktiert und so den Kontakt hergestellt.

B: Herr Denzler: Ja, Mobiliar und Postfinance sind eigentlich auch persönliche Kontakte.

B: Herr Henkel: Wir kannten Mobiliar, die ja mit der Puzzle zusammenarbeitet, die relativ stark im Open Source Umfeld tätig sind.

I: Mit wem arbeitet die Mobiliar?

B: Herr Henkel: Puzzle ITC. Die ist in Bern, ist ein Dienstleister mit 110 Mitarbeitern. Die hatten initial für die Mobiliar eine Software entwickelt. Da ging es um Lima und die Baloise hat gesagt, sie würden es unter der Prämisse, dass es Open Source ist, auch einsetzten. Dann hat die Puzzle praktisch mit der Mobiliar und Baloise die Lima open sourced.

I: Ab welchem Zeitpunkt ist es sinnvoll, mit Partnern zusammenzuarbeiten?

B: Herr Henkel: So früh wie möglich, denn wir können nur von anderen lernen. Einige Firmen sind schon wesentlich weiter als wir. Da sind wir wieder bei dem berühmten Rad, was man nicht neu erfinden muss. In der Open Source Community redet man oft von " Stand on the shoulders of giants". Warum müssen wir die Arbeit nochmals machen, die jemand anderes schon gemacht hat?

B: Herr Denzler: Ja. Der Grund wieso wir uns auch speziell mit anderen Banken und Versicherungen austauschen ist. Das ist sicher auch etwas, wo man voneinander profitieren kann. Damit nicht jeder die kleinsten Paragraphen einer Lizenz abklären muss, sondern dass man ein gewisses Paket mit gutem Gewissen brauchen kann. Und vielleicht auch so Richtlinien oder Best Practices, welche auch Branchenübergreifend anwendbar wären.

B: Herr Henkel: Und es ist natürlich auch denkbar, dass unterschiedliche Anwälte zu unterschiedlichen Entscheidungen kommen. Da hilft es natürlich auch, wenn man Connection zu anderen Firmen hat. Wie die das einschätzten, dass man da zu einem Dialog kommt.

I: Da habe ich jetzt noch ein Thema zu Community. Ihr habt ja gesagt, dass ihr noch nichts veröffentlicht habt und nicht mit einer Community arbeitet. Stelle die Frage mal hypothetisch. Wenn ihr mit einer

zusammenarbeitet wolltet, würdet ihr eine eigene gründen oder einer bestehenden beitreten?

B: Herr Henkel: Es kommt immer darauf an, zu welchem Thema. Ich denke, wenn es um Produkte geht, die schon am Markt sind, die schon eine Community haben, dann werden wir da mitarbeiten. Wenn wir selbst jetzt wie die Baloise den Need haben Open Insurance zu machen, dann würden wir natürlich eine eigene Community gründen und schauen wer kann da an Board kommen. Aber in unserem Umfeld nutzten wir in der Regel kommerzielle Software und entwickeln nicht von Grund auf neu. Wir haben jetzt noch 5 Minuten, ich habe gerade geschaut, der Raum ist dann belegt.

I: Welche Arten von Beiträgen würdet ihr von einer Community erwarten? Oder was erhofft ihr euch von einer Community?

B: Herr Henkel: Denke der kleinste Beitrag ist Issues und Bugs aufmachen. Der nächste Schritt wäre Issue und Bugs fixen. Und der nächste Schritt da drauf wäre, weitere Features zum Produkt hinzuzufügen.

I: Bei Firmen findet die Entscheidungsfindung häufig zentral statt, während in Community eher dezentral entschieden wird. Wie würdet ihr mit diesem Balanceakt umgehen?

B: Herr Denzler: Ja es kommt stark auf die Komponenten drauf an. Also wenn das die Core Versicherungskomponente ist, willst du als Firma schon die Roadmap bestimmen. Aber wenn das mehr technische Sachen sind, dann kann sehr viel von der Community übernommen werden.

B: Herr Henkel: Es kommt darauf an, was der Impact für dich als Firma ist. Bei gewissen Sachen willst du sicher steuern.

I: Und wie würde die Entscheidungsfindung aussehen? Welche Teile vom Code darf zum Beispiel veröffentlicht werden? Wer entscheidet würde da alles mitentscheiden?

B: Herr Henkel: Das Gremium, dass noch zu definieren ist. Keine Ahnung. Wenn wir dann eine Open Source Community haben und an dem Punkt sind, werden wir uns die Gedanken machen. Aber aktuell ist das noch sehr weit weg.

I: Was ist der beste Moment für eine Veröffentlichung von Source Komponenten?

B: Herr Henkel: Der beste Zeitpunkt ist, denke ich, wenn die Grundfunktionalitäten da ist und du dich nicht von Grund auf entwickeln willst, sondern wenn du schon mal was hast, was funktioniert.

B: Herr Denzler: Zentral ist, dass man sich von Anfang an im Klaren ist, dass man es open sourcen will. Das haben wir in der Vergangenheit auch gemerkt. Wenn man nachträglich open sourcen will, was nicht mit dem Gedanken von Open Source entwickelt wurde, dann hast du relativ viel Aufwand. Einerseits ist das der Coder und anderseits die Organisation. Weil du Internals herausnehmen und Sachen umbauen musst. Von dem her, je früher desto besser, aber es muss eine gewisse Maturität schon haben.

B: Herr Henkel: Genau. Ich denke, du musst es frühzeitig in der Planung schon vorhaben zu open sourcen. Du kannst es intern zuerst mal entwickeln, aber musst es frühzeitig vorsehen.

I: Noch eine Frage und dann sind wir mit dem Interview durch. Wie würdet ihr mit Trittbrettfahrer oder mit der Imitationsgefahr umgehen?

B: Herr Denzler: Ja ich denke das ist für uns nicht so relevant. Wie gesagt, wir verdienen unser Geld nicht mit Software. Von dem her ist das für uns nicht matchentscheidend.

Interview mit Michael Stolz und Daniel Zuck am 17.01.2019

I: Das ganze Interview wird aufgenommen und transkribiert. Das Transkript kommt in der Arbeit in den Anhang. Die SIX Group, Ihre Namen und Funktionen werden in der Arbeit erscheinen und gewisse Aussagen zitiert, wenn es gerade zum Thema passt oder was die Meinung in der Praxis zu der Theorie ist. Ist das in Ordnung?

B: Herr Zuck: Ja, das ist in Ordnung.

I: Machen wir eine kurze Vorstellungsrunde. Mein Name ist Vidhushan Asokan und ich studiere an der Universität Bern. Für den Masterabschluss mache ich meine Masterarbeit im Rahmen einer Forschungsarbeit bei Matthias Stürmer. Das Thema ist Open Source Software und ich schaue vor allem Behörden und Unternehmen an, welche in letzter Zeit auch in der Bewegung teilnehmen. Die Frage ist wieso, was und wie machen sie es. Von euch würde ich gerne erfahren, wer ihr seid, was ihre Ausbildung ist, wie ihr zu der SIX Group gekommen seid und was eure aktuelle Funktion ist.

B: Herr Stolz: Ich bin der Mischa Stolz. Ich habe Informationstechnologie in Winterthur an der ZHW studiert. Der Abschluss lautet diplomierter Ingenieur FH in Informationstechnologie. Nach dem Studium war ich sechs Jahre lang in der Softwareentwicklung tätig in einer kleineren Firma. Im Jahr 2007 zur SWX, welche 2008 zusammen mit Telekurs und SIS zum Konzern SIX fusioniert hat. Bei der SIX bin ich als Softwareengineer angestellt. Ich bin in dem Sinne Softwareengineer im Testing, weil ich in einem End-to-End-Testing Team arbeite. In diesem Team mache ich hauptsächlich den Betrieb der Testplattformen. Und auch noch ein bisschen Non-Functional Software Testing. Des Weiteren bin ich bei diesem Interview, weil ich seit ungefähr drei Jahren der Gildenmeister der Open Source Gilde bin, neu heisst es „Head von der Community of Practice Open Source“. Und mich dementsprechend schon länger für den vernünftigen Umgang und Einsatz mit Open Source bei der SIX einsetze. Ich setze auch privat fleissig Open Source ein und als Entwickler in der kleinen Firma damals, war es ganz normal, dass man da keine Berührungsängste hat. Und dort haben wir schon früh Linux eingesetzt.

B: Herr Zuck: Ich bin der Daniel Zuck. Vom Bildungsweg bin ich eigentlich Quereinsteiger mit Betriebswirtschaftsstudium 1990. War aber auch seit 1990 stets in verschiedenen IT-Bereichen tätig. Und die Leidenschaft oder das Interesse an Open Source kommt auch aus dem privaten Umfeld, weil ich dort niemals proprietäre Software eingesetzt habe, ausser es wurde mal zum Gaming gebraucht (lacht). Privat nutze ich ausschliesslich konsequent Free Software oder Linux für alle Aufgaben. Ich bin seit 2011 bei SIX im Linux-Bereich und dort im Engineering-Bereich zuständig für die Plattform, die Konfiguration, Konfigurationsmanagement und allfällige Themen, welche mit der Plattform aufkommen. Das heisst Performance Analyse, Security, Betrachtungen und sagen wir spezielle technische Themen. Und in der Eigenschaft auch sogenannter Techlead für das Linux Team und die Linux Plattform. Darüber eben auch gerne mit Mischa zusammen in der Open Source Gilde, wo wir die Themen Open Source Nutzung in SIX anschauen. Als Plattformsicht werden häufig Fragen an uns gestellt, also an die Linuxplattform. Wo es dann im Kern um Maintenance, Zuständigkeit, Patch, wie sie es in ihrem Fragenkatalog auch antönen und solche Themen geht.

I: Wie ist die Idee bei SIX eigene Projekte oder Teile der Software zu veröffentlichen oder planen zu veröffentlichen und der Allgemeinheit kostenlos zur Verfügung zu stellen?

B: Herr Stolz: Für mich ist der Grund, dass wir in vielen wichtigen Orten Open Source Software einsetzen, aber als Firma offiziell nichts zurückliefern. Das ist in vielerlei Hinsicht natürlich schade. Deshalb finde ich es wichtig, dass wir in Zukunft etwas zurückgeben in dem wir etwas veröffentlichen oder zumindest mit Contributions in einem Projekt uns beteiligen.

B: Herr Zuck: Die persönliche Motivation kommt aus der Überzeugung, dass Knowhow durch das Teilen sich vermehrt. Und praktisch aus eben dem Plattformbereich, das sehr oft Themen wie man es verwendet und wie man es professionalisiert auftreten.

B: Herr Stolz: Aus Businesssicht bin ich der Meinung, dass es paar Bereiche gibt, welche auch den Kunden helfen würden. Wenn man beispielsweise ein API zur Verfügung stellt, wäre es naheliegend, eine Client-Library als Open Source kostenlos für die Kunden zur Verfügung zu stellen.

B: Herr Zuck: Bei Open Source hat man im Zweifelsfall verschiedene Lösungen für das gleiche Problem und kann für sich die passendste Lösung aussuchen. Deswegen muss man nicht unbedingt auf den Marktleader zielen. Also können wir uns für ein Problem, sagen wir ein kleineres Projekt aussuchen und dann auch relativ eng mit dem Projekt zusammenarbeiten und auch Einfluss nehmen. Das ist so meine Vision, meine Zielvorstellung mit dem Zurückgeben, dass man als Unternehmen einzelne Features in kleineren Projekten beeinflussen kann. Ich sage schon einschränkend kleineren Projekten. Etwas wie beispielsweise Apache HTTP, PHP oder Python sind natürlich Projekte von einem anderen Kaliber. Wir als SIX, bei welcher das Kernbusiness Software schon irgendwo ist, aber jetzt nicht die Softwareentwicklung in Projektmanagement Sinne oder Veröffentlichung Sinne. Eher der Servicebetrieb ist unser Kernbusiness. Bei so grossen Projekten besteht natürlich keine grosse Möglichkeit jemals Einfluss zu nehmen.

I: Wie und wann ist die Open Source Gilde bei SIX entstanden?

B: Herr Stolz: Ich bin der Meinung, dass die Idee und die Initiative von einem einzelnen Mitarbeiter kamen, welcher mittlerweile in unserem Innovationsabteilung F10 arbeitet. Das war, glaube ich, gleichzeitig mit der Zentralisierung der IT. Das ganze Engineering für die Börsenplattform war früher bei der Business Unit und spaltete sich dann von der Business Unit ab, sodass alle IT Mitarbeiter in einer zentralen IT Abteilung gelandet sind. Und zu diesem Zeitpunkt hat man natürlich über Synergien nutzen und Erfahrungsaustausch nachgedacht. Dann hatte jemand die Idee der Gildensysteme, in welcher ein Erfahrungsaustausch zu bestimmten Fachthemen über einzelne Teams und Bereiche hinweg stattfinden sollte. Und er hat damals auch paar Themenvorschläge gebracht. Eine Gilde war dann die Linux Gilde, welche ein Jahr lang vakant war und sich niemand um die Gilde gekümmert hat. Ich habe dann die Gilde übernommen, jedoch fokussiere ich mich nicht nur auf Linux, sondern es soll allgemein eine Open Source Software Gilde werden.

I: In welchem Jahr war das?

B: Herr Stolz: Ich bin der Meinung, es war 2016

B: Herr Zuck: Also es gibt noch eine C++, Python, Docker Gilde. Es hatte mal eine Softwaretesting Gilde, stoss jedoch nicht so auf grosses Interesse.

B: Herr Stolz: Die ist wieder eingeschlafen. Am Anfang gab es so Präsentationsevents, in welchen Informationen an ein grösseres Publikum übermittelt wurden, aber wenig interne Konvolution gehabt. Der Gildemaster empfand das nicht als ausreichend und hat die Aktivität eingestellt. Seither hat das niemand übernommen.

I: Die Open Source Gilde gibt es heute noch?

B: Herr Stolz: Ja, die gibt es noch.

I: Über die interne Organisation kommen wir später noch zu sprechen. So wie ich das verstehe, entstanden Gilden durch Zentralisierung, weil Synergien geschaffen und die Kommunikationswege kürzer wurden?

B: Herr Stolz: Grundsätzlich war es eine Idee von einer Person, aber offensichtlich auf Anklang bei mehreren gestossen. Dann hat man das in einem sehr freien Rahmen umgesetzt. Es wurde kein explizites Budget dafür genehmigt.

B: Herr Zuck: Da muss man auch noch betonen oder erklären, wie die SIX als grösseres Unternehmen funktioniert. Dass verschiedene Themen eben in verschiedenen Teams sind. Und die Teams waren natürlich sehr stark auf sich selbst gerichtet. Das heisst, dass bei einem gegebenen beliebigen Problem im Haus, im Zweifelsfall mehr als eine Lösung entwickelt wird. Parallel das gleiche Problem mehrfach gelöst wird. Die Sichtbarkeit von teamübergreifenden Skills schlecht ist. Beispielsweise fehlt die Transparenz über die Affinität der Betriebssysteme. Wer ist Linux bzw. Windowsaffin, erschliesst sich zum Teil nicht mehr über die Teamfunktion. Wenn es nicht offensichtliche organisatorische Indizien hat, welches Skills man hat, ist es unter den Mitarbeitern nicht so transparent. Und das aufzuweichen, das ist so die Idee der Gilde. Dass man nicht nur im Team und dem Blick auf sich selber arbeitet, sondern auch zwischen den Teams arbeitet. Networking ist da das Stichwort. Das Zweite ist das Arbeitsgefäss, sozusagen das Buchungsgefäss. Unsere Arbeitszeit muss rapportiert werden und wenn wir da keinen akzeptierten Grund haben, ist es unproduktive Zeit. Die ist dann für die persönlichen Ziele blöd. Da ist es schön, dass es dann für die Gilde ein offizielles Gefäss gibt,

also dass wir die Zeit für Open Source Themen im Sinne des Unternehmens nutzen, auch in unseren persönlichen HR Performance Reporting als produktive Zeit auffallen kann.

I: Ihr habt gesagt, dass keine expliziten Ressourcen zur Verfügung gestellt wurden, aber durch die Bottom-up Initiative das Management ein Buchungsgefäss eröffnet hat, in welcher man die Arbeitszeit anrechnen konnte?

B: Herr Zuck: Genau.

B: Herr Stolz: Spezifisch für Leute, welche eine Gilde als Master betreiben, konnte man immer auf eine gewisse Kostenstelle verbuchen, welches relativ weit oben im Application Engineering Management angesiedelt war. Also sehr Softwaredevelopement lastig in dieser Hinsicht.

I: Hattet ihr für die Genehmigung dieser Arbeitszeit Schwierigkeiten? Wie ist der Prozess abgelaufen?

B: Herr Stolz: Das lief informell. Man konnte sich melden und an Themen arbeiten. Wenn man irgendwie Geld benötigte, musste man das spezifisch anfragen.

I: Und für die Eröffnung des Buchungsgefässes?

B: Herr Stolz: Das weiss ich natürlich nicht, weil ich nicht beteiligt war.

I: Habt ihr schon Projekte, Software oder Teile der Software veröffentlicht oder planen sie es zu tun?

B: Herr Zuck: Aus Unternehmungssicht haben wir als SIX nichts veröffentlicht. Sicherlich haben aber SIX Mitarbeiter privat etwas veröffentlicht. Das ist sozusagen die Knacknuss, die sich stellt. Nämlich auf Basis von früheren Compliance-Richtlinien das Veröffentlichen explizit verboten wurde und das sitzt relativ tief in den Köpfen drin. Als Techniker und Entwickler findet man da aber Umgehungen, welche viele Mitarbeiter teilweise auch genutzt haben. Dass man dann auf GitHub mit einem privaten Account arbeitet.

I: Das wäre so eine Umgehung?

B: Herr Zuck: Das wäre beispielsweise eine Umgehung. Das wäre aus Unternehmungssicht unvorteilhaft, weil der Mitarbeiter vom Unternehmen nicht geschützt ist oder er keine klaren Richtlinien hat, wie er bei der Veröffentlichung vorgehen sollte, damit es für alle Beteiligte vorteilhaft ist. Also zum Beispiel, welche Lizenz er nutzen soll, wie soll der Veröffentlichungsprozess sein und wie behandelt man das Thema von lost intellectual property, weil das im Zweifelsfall gegen bestehenden Weisungen verstösst. Das ist so diese Grauzone und diese Nuss zu knacken, ist Teil der Challenge zurzeit. Das Ziel ist diesbezüglich Richtlinien zu haben, was die Veröffentlichung von Code angeht. Das fängt bei Kleinigkeiten an, wie zum Beispiel das Schreiben eines Patches für das Projekt X und stelle dann den Patch online. Dann bin ich im Bereich, dass ich Code veröffentliche, den ich als SIX Mitarbeiter geschrieben habe. Und in meinem Arbeitsvertrag steht klar drin, dass die exklusiven Nutzungsrechte des Codes dem Arbeitgeber zufallen. Wenn ich da keine passende Lizenz habe, könnte es dann mit der Weiterverwendung schwierig werden.

I: Also aktuell werden keine Projekte unter dem Namen SIX Group veröffentlicht?

B: Herr Stolz: Gemäss unserem Wissen nicht.

I: Und das Ziel von der Gilde wäre, dass veröffentlicht werden sollte?

B: Herr Stolz: Ich sehe ganz konkret denAnwendungsfall API, weil wir eine Strategie fahren, dass wir den Businesskunden ein API zur Verfügung stellen, welche sie nutzen können. Dort wäre es super, wenn man eine Referenzimplementierung als Open Source zur Verfügung stellen würde. Es würde dem Kunden vieles vereinfachen. Man könnte es aber auch selbst nutzen, weil man die Referenzimplementierung auch beim Testing des API einsetzten könnte. Ich habe die Hoffnung, dass wir auch mal die richtigen Businessleute ins Boot holen können, welche diesen Nutzen sehen und verstehen, wie sie damit profitieren würden. Wenn es einfacher ist für den Kunden, weil er sich via eines fertigen Client Library sich auf das API onboarden kann. Dann kommen auch mehr Kunden, weil je einfach desto besser. Ich sehe hier immer den Vergleich mit PayPal. Wenn man sich für die Zahlungsabwicklung an PayPal anbinden will, dann findet man im Internet auf der PayPal Seite jede Menge von Dokumentationen und Example Code. Ich weiss jetzt nicht genau ob es auch eine Client Library gibt. Aber es wird jedenfalls dem Entwickler leicht gemacht, sich anzubinden. Das Gleiche kenne ich bisher nicht, dass es von der SIX angeboten würde,

kann jedoch durchaus sein, dass es im Rahmen von API Aktivitäten vorgesehen ist. Für das sind wir beide zu weit von den Bereichen weg, um eine Aussage dazu zu machen.

B: Herr Zuck: Wir haben schon mit unserem CIO gesprochen. Da besteht bezüglich Veröffentlichung ein Konsens. Den brauchen wir gar nicht zu bearbeiten, sondern er sieht das klar auch. Er sieht auch diese Grauzonen und sieht auch, dass im Sinne von Mitarbeiterschutz das Unternehmen diese Grauzonen auflösen bzw. regeln sollte. Ja in diesem Umfeld bewegen wir uns.

I: Vorher haben sie unter anderem ihre private Motivation erläutert. Wenn wir nur die SIX anschauen, was wären Chancen und Vorteile für SIX, eigene Projekte zu veröffentlichen?

B: Herr Zuck: So Sachen wie spezielle Libraries, QuickFix. Ein Beispiel, wo es um Finanznachrichten geht und die Library behandelt Finanznachrichten. Desweiteren verwenden wir intern das SaltStack Configuration Management, welches jetzt nicht so das grosse Projekt ist. Das wäre ein Kandidat, bei welchem wir auch Einfluss nehmen könnten, wenn wir dort geregelt Code veröffentlichen könnten. Oder ein anderes Projekt X2Go als X-Client für Windowssysteme, welches relativ verbreitet verwendet wird. Bei diesem relativ kleinen Projekt könnte man es in unserem Sinne unterstützen. Das sind so die naheliegenden Kandidaten. Die Motivation von Plattformseite ist, dass man etwas geregelt veröffentlichen kann. Umgekehrt wäre auch der geregelte Einsatz eine Motivation, weil man als Entwickler heute keine Guideline hat, was zum Beispiel der Einsatz von AGPL Projekten angeht. Dass man da im Zweifelsfall bei Lizenzierungsthemen aufpassen müsste.

I: Haben sie noch etwas anzufügen?

B: Herr Stolz: Eine Motivation könnte auch sein, dass wir als Unternehmen attraktiver werden. Recruiting ist ein wichtiges Thema für uns. Speziell die guten Entwickler zu kriegen ist heutzutage nicht einfach. Ich persönlich bin überzeugt davon, dass wenn wir im Open Source Bereich mit eigenen Projekten und Software präsenter wären, dann hätten wir definitiv gute Chancen gute Leute darüber zu rekrutieren.

B: Herr Zuck: Kann ich bestätigen, guter Punkt. Gestern gab es ein Gespräch über Recruiting mit meinem Chef, weil er häufig aktiv Leute rekrutiert. Und er bestätigte mir, dass SIX als das Amt für Datenverarbeitung wahrgenommen wird. Als relativ staubig. Intern ist es jedoch gar nicht so, denn wir machen solche Initiativen jetzt aus eigenem Interesse und mit einer gewissen Nachhaltigkeit. Offensichtlich kommen wir ja voran. Du sitzt hier und wir dürfen über das Thema reden. Das bewegt sich etwas in eine Richtung, welches noch nicht gerade wie ein kleines Start-up ist, aber es bewegt sich etwas. Das ist die entscheidende Botschaft.

I: Wie sieht es mit Kosten aus? Wäre dies auch eine Motivation?

B: Herr Stolz: (..). Wir haben natürlich einen Kostendruck. Für viele Leute wäre es eine gute Motivation, wenn man damit Kosten sparen könnte. Es ist natürlich nicht wie früher, wo man blauäugig geglaubt hat, dass mit Open Source, irgendwelchen Veröffentlichungen und Zusammenarbeit mit Communities man Geld gespart hat. Mittlerweile weiss man auch, dass ein Communityaufbau oder ein Projekt öffentlich zu betreiben einen Aufwand braucht. Man muss die Qualität gewährleisten, weil man unter anderem einen Imageschaden hat, wenn man Sachen veröffentlicht, die nicht qualitativ hochwertig sind. Von dem her gesehen glaube ich, dass der Kostenfaktor eher im Hintergrund steht.

B: Herr Zuck: Respektive, was kostet eine Herstellerabhängigkeit? Da gibt es auch einige Bereiche, die da erwachen. Dass die Bindung an einen Hersteller für eine Lösung im Zweifelsfall auch nicht so das Wahre ist. Also dass da in Hochglanzprojekten- und -Prospekten mehr versprochen als gehalten wird, in verschiedenen Toolingbereichen zum Beispiel. Solche Enterprise Tools rufen natürlich von Haus aus eine gewisse Preisklasse aus. Im Vergleich dazu eigenes Know-How aufzubauen, welches es mit Open Source attraktiv macht. Es gibt sicher dann irgendwo einen Mittelweg, bei dem beides relevant ist. Also, dass die Integrationsleistung, die ein Open Source Hersteller leistet, für uns von Vorteil ist. Zum Beispiel Red Hat mit Enterprise Linux oder OpenShift, wo verschiedene Open Source Projekte mit einer guten Qualität ins Haus kommen. Da geben wir indirekt durch die Supportpreise an Red Hat etwas nach aussen. Aber es auch flexibler handhaben zu können, ist dann das Ziel. Dass es nicht über so einen Anbieter kanalisiert wird, sondern wir auch selbst direkt als SIX in der Öffentlichkeit tätig werden können.

I: Sie haben vorher erwähnt, dass es im Management erkannt wurde, dass es Sinn machen kann. Gibt es eine Open Source Software Strategie und ist sie in die Firmen- oder IT-Strategie implementiert?

B: Herr Zuck: Das ist eine gute Frage. Das sind Bereiche, wo wir darüber stolpern und jovial gesagt Staub aufwirbeln. Mit wem vom Management spricht man und wie relevant ist es für das jeweilige Management? Also das ist es für ein Business Management von Payments oder Börse, die interessiert Technik, vorsichtig ausgedrückt, gar nicht. Das sieht jetzt beim Management von unserer IT Division vollkommen anders aus. Also im SIX Aufsichtsrat hätte man wahrscheinlich Schwierigkeiten das Thema zu vermitteln. Einfach wegen der Ferne zu den technischen Themen. Dass da das nötige Verständnis nicht da ist. Das ist vielleicht ein Punkt, in welcher SIX sich ausfinden und einlöten muss, bis in welche Bereiche muss im Management Verständnis über eher technisch geprägte Aspekte da sein. Unsere Wahrnehmung im Gespräch mit dem Management der IT Division ist, dass die Begriffe, das Verständnis und das Ziel klar sind. Das nötige Interesse, an sowohl Open Source als auch proprietären Lösungen, ist auf jeden Fall da. Der Bereich Open Source Strategie ist sicherlich etwas, wo wir idealerweise am Ende von unseren Aktivitäten stehen. Der Bezug, das Zurückgeben an Projekte oder das Veröffentlichen Bestandteil von der Strategie wird. Jetzige IT Strategie ist sehr High-Level und dort tauchen spezifische Lösungen im Sinne von proprietär oder Open Source gar nicht auf.

B: Herr Stolz: Mit einer kleinen Ausnahme, oder? Es gibt doch diese IT Strategie, in welcher zumindest steht, dass man im Plattformbereich Open Source gegenüber proprietären Lösungen Vorrang geben sollte, wenn sie gleichwertig sind.

I: Also bei der Nutzung?

B: Herr Stolz: Ja, das ist bei der Nutzung.

B: Herr Zuck: Da muss man ergänzend dazu sagen, dass diese Strategie in der Vernehmlassung- und Einsetzungsphase ist. Was auch wiederum reflektiert, wo und wie sich SIX gerade bewegt. Nämlich, dass auf der nötigen High-Level Ebene die Punkte verankert werden und dass wir als Gilde so ein bisschen Graswurzelarbeit noch machen.

I: Sprechen wir mal über diese Gilde. Wie viele Mitglieder umfasst die Gilde?

B: Herr Stolz: Weil es bis vor zwei Monate keine Pflicht gab Mitgliederlisten zu führen, kann ich im Moment nicht sagen wie viel es sind. Ich habe eine E-Mail Verteilerliste mit ungefähr 50 Personen, welche ich über Aktivitäten informiert habe. Ich kann auch sagen, dass der erfolgreichste Event bzw. eine Präsentation über 100 Personen von der ganzen Firma anzog. Aber ein typischer Teilnehmerrahmen ist eher bei 30 Personen.

I: Wie viele setzen sich aktiv mit Open Source Themen auseinander?

B: Herr Stolz: Aus der Gilde sage ich mal fünf.

I: Da gehören sie zwei dazu und noch drei andere?

B: Herr Stolz und Herr Zuck: Ja

I: Was für Rollen gibt es in der Gilde?

B: Herr Stolz: Es gibt eigentlich nur der Head und Members. Mehr ist da noch nicht aufgeteilt. Ist auch von den Regeln nicht vorgesehen. (...) Ah gut, es gibt noch einen Sponsor (lacht).

I: Ist dieser vom Business Bereich?

B: Herr Stolz: Nein, er ist einfach vom Management und zwar von einem gewissen Level.

I: Wie fähig ist er, ihre Anliegen bei den anderen Management Kollegen einzubringen?

B: Herr Stolz. Weil der Sponsor auch erst seit zwei Monate ein Erfordernis ist, ist er im Moment noch nicht festgelegt.

I: Also es ist noch nicht bekannt, wer das sein wird?

B: Herr Stolz: Nein.

I: Aber es ist geplant?

B: Herr Stolz: Ja. Könnte momentan niemanden nennen. Ich brauche zuerst einmal die Zustimmung von jemandem (lacht).

I: Seit zwei Monate hat sich etwas geändert, haben sie vorher erwähnt?

B: Herr Stolz: Ja genau, also für die Communities of Practice, wie es jetzt heisst, gibt es klare Regeln, Budget pro Jahr und einen maximalen nicht speziell zu genehmigendem Zeitaufwand, welche die Mitglieder für die Gildearbeit buchen können, ohne dass sie ein extra Approval benötigen. Das sind 50 Stunden pro Jahr.

I: Pro Person?

B: Herr Stolz: Ja, pro Mitarbeiter. Also das heisst, wenn man eine Gilde mit vielen Mitgliedern wäre, könnte man verstreut über alle, viel Aufwand einsetzen. Aber in der Realität wird das wohl nicht so passieren. Die meisten Aufgaben kann man nicht so parallelisieren.

I: Jeder Mitarbeiter kann freiwillig beitreten?

B: Herr Stolz: Ist natürlich freiwillig, genau.

I: Je attraktiver eure Gilde ist, desto mehr Ressourcen hättet ihr indirekt ohne Genehmigung zur Verfügung?

B: Herr Stolz: Könnte man so sagen.

I: Könntet ihr auch mehr als die 50 Stunden aufwenden? Wer müsste das genehmigen?

B: Herr Zuck: In der Vision ja. Weil so etwas wie die 10% Regelung, wie Google es handhabt, bei uns als Vision da ist. Also das durch verschiedene Aktivitäten, die entlastend sein sollen. Am Ende dann der Mitarbeiter Freiraum von 10 % hat, die er dann einsetzen kann für Aktivitäten. Und da kann natürlich die Arbeit für die Gilde auch mit dabei sein.

I: Ist es eine Vision oder wird es schon gelebt?

B: Herr Zuck: Es ist eine Vision, es ist auf der Agenda. Aus Ressourcengründen wird es momentan nicht gelebt.

B: Herr Stolz: Davon habe ich sogar noch gar nichts gehört. Ist das mehr bei euch ein Thema?

B: Herr Zuck: Ja das ist mehr bei uns.

B: Herr Stolz: Ah, also bei uns ist es gar kein Thema bisher.

I: Kommen wir zu Compliance und Richtlinien. Habt ihr eine offizielle Open Source Compliance oder Richtlinien?

B: Herr Stolz: Eine aktuelle haben wir nicht. Wir haben eine uralte und die ist eher restriktiv ausgelegt. Es könnte eigentlich gar niemanden einen Source Code veröffentlichen, ausser vielleicht mit der Genehmigung von ganz Oben. Das wäre wirklich auf der allerhöchsten Ebene oben.

I: Erarbeitet die Gilde Compliance und Richtlinien für Open Source?

B: Herr Stolz: Ja, wir arbeiten aktuell daran. Deswegen haben wir Kontakte mit anderen Firmen aufgenommen, welche in einer ähnlichen Entwicklungsstufe wie wir sind. Wir erhoffen uns, dass wir ein paar Anhaltspunkte finden, wie so eine Richtlinie aussehen könnte.

I: Welche Prozesse bräuchte es dann für die Entwicklung, Veröffentlichung und Wartung von Open Source Software? Sie haben vorher Arbeiterschutz genannt. Aber wie würde dieser Prozess aussehen, dass man sowohl den Arbeitnehmer als auch den Arbeitgeber schützen könnte?

B: Herr Stolz: Das ist die schwierige Frage. Das ist, was eben noch zu definieren wäre. Höchstwahrscheinlich muss es mal eine Art von Genehmigungsprozess geben. Es müssen sicher mal Legal-Leute beteiligt sein. Es wird sich zeigen, ob es in dem Sinn mehr um Richtlinienausarbeitung und Verhaltensanweisungen geht oder ob sie in einem konkreten Fall beteiligt sein müssen.

I: Haben sie schon Legal-Leute involviert?

B: Herr Stolz: Ja.

B: Herr Zuck: Also es wird sicher eine Art Freigabeprozess geben, wo es ähnlich wie bei Budgetthemen irgendein selbst zu entscheidenden Freigabeprozess gibt. Also wenn es um kleine Patches geht, dass man da in gewissen Rahmenbedingungen veröffentlichen kann. Wenn es dann um aktives Steuern von Projekten, Eingreifen in Projekte oder ein ganzes Projekt zu veröffentlichen geht, da braucht es sicher noch einen zu definierenden Freigabeprozess.

I: Wo sehen sie denn den grössten Bedarf für die Optimierung von Open Source Entwicklung und Veröffentlichung? Was

sind eure nächsten Schritte oder aktuelle Schwerpunkte?

B: Herr Zuck: Der schwierigste Punkt ist bei der gegebenen Unternehmensgrösse von SIX den aktuellen Startpunkt zu manifestieren. Wo stehen wir in verschiedenen Bereichen? Was ist überhaupt vorhanden? Dieser Startpunkt zu manifestieren, das ist eine praktische Schwierigkeit.

I: Wie viele Mitarbeiter hat die SIX?

B: Herr Zuck: International sind es jenseits der 1000 in der Softwareentwicklung. Ich weiss jetzt nicht wieviel es in der Division sind, aber schon ein paar hundert.

I: Sind in der IT 1000 Mitarbeiter beschäftigt?

B: Herr Zuck: In der IT n-hundert.

B: Ja n-hundert. Es sind unter 1000.

B: N-hundert, sagen wir mal zwischen 400 und 700. Weiss es nicht genau. Da sind natürlich viele Entwickler und viele Geschmacksrichtungen. Die Herausforderung ist, die entsprechend in irgendeiner Form einzufangen und mit einem guten Konsens zu kommen, der dann auch im Unternehmen eingesetzt werden kann.

I: Wo sehen sie momentan die grössere Schwierigkeit. Das Management oder die Entwickler zu überzeugen?

B: Herr Zuck: Bei den alteingesessenen Mitarbeitern, würde ich jetzt mal so sagen.

B: Herr Stolz: Es kann sowohl Management als auch die Entwickler sein. Es gibt Entwickler, welche für sich keinen Nutzen sehen und es gibt aber auch Manager, welche den Business Value nicht sehen.

I: Versucht die Gilde nun einen Business Value auszurechnen und darzulegen?

B: Herr Stolz: So weit sind wir noch nicht.

I: Ihr nutzt viel externe Open Source Software. Haben sie da Richtlinien oder Tools, welche ihr verwendet und die vorgeben, wie man externe Komponenten verwenden darf?

B: Herr Stolz: Wir haben sicher gewisse Richtlinien oder sagen wir mal Best Practices, welche sich etabliert haben. Ich glaube aber, dass sie nicht so einheitlich und nicht unbedingt allen bekannt sind. Mittlerweile ist es so, dass wir nicht einfach Sachen downloaden können. Dementsprechend muss man im Moment bei einer zentralen Stelle die Bereitstellung anfragen, dass es sie einem im internen Repository bereitstellen. Was ich jetzt nicht weiss ist, inwiefern die Zentrale eine Richtlinie habt, was sie bereitstellen dürfen und was nicht. Das ist wirklich ein wenig unklar. Das ist wirklich einer der Gründe, warum wir stärker an diesen Grundlagearbeiten sind und wir auch einheitliche Richtlinien für die Nutzung haben möchten. Ursprünglich haben wir gedacht, dass diese schon eher existieren und wir eher an Richtlinien für die Veröffentlichung arbeiten könnten. Fakt ist, dass wir noch keine einheitliche Richtlinie für die Nutzung haben.

I: Unter welcher Lizenz würdet ihr eigene Projekte veröffentlichen?

B: Herr Stolz: Da bin ich nach wie vor der Meinung, dass es immer vom spezifischen Projekt abhängt. Das müsste man eigentlich mit einem entsprechenden Business Owner abstimmen. Je nachdem was es für ein Business ist, kann es durchaus sein, dass eine Copyleft Lizenz von Vorteil wäre. Dann könnte nicht irgendein Konkurrent es nehmen, damit ein neues Produkt bauen und es dann nicht mehr veröffentlichen. Umgekehrt kann es natürlich auch sein, dass wir den Kunden ermöglichen wollen, dass sie ein Projekt darauf aufbauen können, welche sie nicht zwingend veröffentlichen müssten. Dementsprechend würden wir eine permissive Lizenz ohne Copyleft nehmen. Meine Meinung ist, dass es wirklich abhängig vom spezifischen Projekt und Businessnutzen ist, welchen man sich erhofft. Kann man nicht so generisch entscheiden.

I: Haben sie schon Erfahrung mit Lizenzkonflikten gehabt?

B: Herr Stolz: selbst nicht. Problematik ist bekannt. Du vielleicht, hattet ihr schon mal konkret Konflikte?

B: Herr Zuck: Konkrete Konflikte, nein. Mit kommerziellen Lizenzen welche eingesetzt werden, aber mit Open Source noch nicht. Aber es hat sicher das Potential, dass diese Themen kommen werden, wen APIs veröffentlicht werden oder öffentlich bereitgestellt werden. Viele Kollegen in der Entwicklungsseite haben diverse Open Source Komponenten einbezogen, haben halt vorwärts gearbeitet,

weil bezüglich time to market schon eine Relevanz hat, da schnell zu arbeiten. Umgekehrt, wie Mischa schon erwähnt hat, bei den Strategie- und Regelungsthemen nicht in der nötigen Geschwindigkeit nachgekommen sind. Da sind wir jetzt am Aufholen.

B: Herr Stolz: Zudem haben wir bisher, gemäss unserem Wissen, gar nie etwas veröffentlicht, sondern alles war für die interne Nutzung. Das Maximum was nach aussen gelang, war das man ein Service zur Verfügung gestellt hat, womit eigentlich klar ist, dass solange keine Lizenz wie zum Beispiel die AGPL irgendwo verbaut wurde, keinerlei Probleme hat. Man hat sich das gar nicht überlegen müssen, weil alles nur innerhalb der Firma ausgebaut wurde.

I: Also habt ihr auch keinen Prozess für Konflikte?

B: Herr Stolz: Nein. Wie gesagt, die Lizenzkonflikte kommen ja definitiv dann zum Tragen, wenn man anfängt zu veröffentlichen.

I: Musstet ihr schon mal was Zwangsveröffentlichen?

B: Herr Stolz: Nein, da wir eigentlich gar nie Software vertreiben. Twint ist ein Tochterunternehmen zwischen PostFinance und SIX zusammen. Bei diesen Leuten müsste man jemanden finden, der sich mit der Thematik auskennt, hoffe ich.

I: Wie werden die Mitarbeiter im Umgang mit Open Source geschult?

B: Herr Zuck: Bisher relativ wenig. Das sehe ich bei uns bei der Gilde auf der Agenda, dort auch Rahmenbedingungen zu geben. Nach dem Feedback von Entwicklern sind die groben Sachkenntnisse da. Das reicht in der Perspektive ja fast aus, auf gewisse Lizenztypen hinzuweisen und welche für welchen Use Case passen. Dass dann die Schulung auf die Veröffentlichung und Richtlinie abzielt.

I: Wie kann man das Lizenzrecht den Entwicklern spannend erklären?

B: Herr Stolz: Die aktuelle Idee ist, dass man eine Tabelle von den Lizenztypen mit Kategorien von den üblichen Anwendungsfälle macht, sodass sie anhand von dem entscheiden könnten, welche Lizenzen in Frage kommen und welche nicht. Nach der Kategorie interne Nutzung, Service zur Verfügung stellen oder allenfalls dann Software verteilen. Anhand von dem könnte man es dann entscheiden, ob es besser ist etwas einzubauen oder nicht einzubauen.

I: Wie würdet ihr externe dazu motivieren bei ihren Projekten oder Vorhaben mitzumachen?

B: Herr Stolz: Im Moment ist das eine sehr hypothetische Frage.

B: Herr Zuck: Eben wir veröffentlichen ja noch nichts, deswegen ist es ja eine hypothetische Frage. Die können wir momentan nur verdichten auf die Recruiting Frage, indem wir eben betonen, dass SIX intern nicht unbedingt das Amt für Datenverarbeitung ist, sondern dass wir intern solche Initiativen starten, ernsthaft betreiben können und weiterkommen. Das ist auch eine Wahrnehmung von mir in anderen Bereichen. Wenn jemand wirklich etwas zu sagen hat und mit dem nötigen Kontext kommt, dass das dann auch gehört wird.

B Herr Stolz: In Anbetracht fortgeschrittener Zeit aber vor allem können wir diese Themen gar noch nicht richtig beantworten, weil wir nicht so weit sind. Wenn du noch andere Fragen hast, dass wir langsam gegen den Schluss gehen.

I: Arbeitet ihr mit Partner zusammen in Bezug auf Open Source Veröffentlichung?

B: Herr Zuck: Nein, nicht direkt. Sicher arbeiten wir zum Beispiel mit Red Hat zusammen oder anderen Projekten, wo wir eine Lizenz beziehen. Aber ich unterstelle mal, dass das ist nicht die Partnerschaft meint, die du da fragen willst.

I: Ab welchem Zeitpunkt wäre es gut einen Partner ins Boot zu holen?

B: Herr Zuck: Wenn das Business Interesse hat, etwas in einem Projekt zu ändern, dann ist es ein guter Zeitpunkt. Wenn das Business es erkannt hat, dass sich hier etwas bewegen und mein Interesse einbringen lässt.

I: Noch zwei Fragen habe ich noch. Wie ihr gesagt habt, seid ihr noch nicht so weit, dass ihr ein Community Management betreibt. Aber was würdet ihr von einer Zusammenarbeit erwarten oder was erhofft ihr euch von der Community?

B: Herr Stolz: (...) Ein Vorteil von der Zusammenarbeit mit der Community wäre, dass man auch noch Entwickler dabeihätte, welche eine andere Perspektive einnehmen könnten, als unsere Entwickler innerhalb von der Firma haben. Vielleicht können sie auch vielmehr die Sicht eines Kunden oder Nutzers einnehmen bzw. es fällt ihnen leichter. Oder sie haben andere Skillsets, wissen wie Sachen anderswo entwickelt werden oder wie man zusammenarbeitet, was wir so nicht kennen. Weil wir vielleicht seit 10 Jahr anders arbeiten. Glaube, das könnte viel Input bringen, dass wir andere Entwicklungsmodelle live sehen und nicht nur auf dem Papier, sondern von Leuten es erfahren, welche anders schon gearbeitet haben.

B: Herr Zuck: Oder auch ein Ansporn an sich selbst, den Code vernünftig zu entwickeln. Dass man als Entwickler das benötigte Selbstbewusstsein entwickelt und hat, auch den Code zu veröffentlichen. Wenn ich weiss, mein Code bleibt in meinem Kämmerlein und wenn ich den übersetzt habe, dann sieht ihn niemand mehr. Dann kann ich da auf Deutsch gesagt "Rumsauen". Also eine Qualitätssteigerung.

I: Wann ist der beste Moment für eine Veröffentlichung?

B: Herr Zuck: Bei einer Entwicklungsphase wo man schon ein Teil der Funktionalität sieht, also ein Prototyp.

B: Herr Stolz: Vielleicht so ein Minimum Viable Product.

I: Noch die letzte Frage wäre über sogenannte Trittbrettfahrer und Imitationsgefahr. Wie schätzen sie diese Problematik im Zusammenhang einer Veröffentlichung ein?

B: Herr Stolz: Aus meiner Börsensicht würde ich sagen, ist die sehr klein. Der Börsenbetrieb ist extrem reguliert und es ist nicht nur ein technisches Thema eine weitere Börse aufzubauen, sondern auch ein regulatorisches. Wenn wir einen direkten Konkurrenten kriegen sollten, also es gibt natürlich schon einen, die Berner Börse. Es ist eben nicht wirklich ein direkter Konkurrent, weil die haben einen ganz anderen Marktanteil innerhalb der Schweiz. Wenn wir direkte Konkurrenten hätten, sind dies andere Internationale Börsen, aber die sind dann wieder anders reguliert. Die kommen auch nicht so auf die Idee in der Schweiz eine zweite Börse aufzubauen. Ist meine persönliche Meinung, aber ich glaube dort haben wir mit der Börsen-IT nicht viel Gefahr.

B: Herr Zuck: Mein persönlicher Eindruck ist, dass ich da auch keine Gefahr sehe. Eher den Nutzen von den anderen Perspektiven und Punkten die wir schon erörtert haben. Sachen wie Trittbrettfahrer und Loss of intellectual property nehme ich eher in Bereichen wahr, die mit Technik nichts zu tun haben. Wo die Idee von Open Source eher weniger bekannt ist. Eher Business lastige Bereiche, wo dann allfällige Themen und Fragen aufkommen.

I: Somit wäre ich mit den Fragen durch. Bedanke mich sehr für das Gespräch, war sehr spannend und ausschlussreich mal in die Praxis einzusehen.

Interview mit Thomas Joos am 25.01.2019

I: Das Interview wird semi-strukturiert geführt. Zuerst würden mich ein paar persönliche Informationen interessieren. Was ist ihre Ausbildung, wie sind Sie zum KAIO gekommen und was ist ihre Funktion beim KAIO?

B: Ich komme ursprünglich aus einem ganz anderen Bereich, das ist jedoch schon sehr lange her. Ich bin vor ungefähr 30 Jahre als Quereinsteiger in die Informatik gekommen, dazumal in der Bundesverwaltung. Ich habe dann ungefähr 20 Jahre im Bereich SAP gearbeitet. Als Projektleiter und Berater auf kantonaler und Bundesebene, aber auch international bei einer deutschen Softwarefirma. Ich bin jetzt seit vier Jahren beim KAIO und bin verantwortlich für Themen wie eben zum Beispiel Open Source, für das Bewirtschaften von Fach- und Konzernapplikationen. (über den ganzen Lifecycle, von der Beschaffung bis zu der Einführung, Wartung und Betrieb) Wir denken nicht mehr in Applikationen, sondern in Services, welche wir unseren Kunden zur Verfügung stellen. Zum Beispiel bestellt jemand den Artikel „Benutzerkonto“; das KAIO zeichnet sich dann verantwortlich für alle notwendigen Aktivitäten, damit das Benutzerkonto funktioniert. IT-technisch macht der Kanton heute nichts mehr selbst: Wir haben alles an externe Dienstleister ausgelagert. Das heisst, dass wir keine Systeme mehr selbst betreiben. Wir bewegen uns mehr auf einer übergeordneten Ebene mit der Koordination von Lieferanten und mit unseren Anspruchsträgern.

I: Welche Lieferanten sind für das KAIO tätig?

B: Der Kanton hat sehr viele Lieferanten, welche für uns Leistungen erbringen. So haben wir zum Beispiel im Bereich Telefonie die Swisscom. Bei den Netzwerken ist dies die SPIE ICS. Für den Betrieb unserer Systeme ist vor allem die Bedag zuständig. Daneben gibt es natürlich noch viele kleinere Lieferanten, welche Dienstleistungen für den Kanton erbringen.

I: Die Softwareriesen sind da nicht dabei? Wie zum Beispiel Microsoft?

B: Doch, selbstverständlich ist Microsoft auch ein Lieferant von uns. Der Kanton setzt im Bereich Office die Produkte vom Microsoft ein. Es ist irgendwie so selbstverständlich, dass ich nicht an Microsoft gedacht habe.

I: Für unsere Studie haben wir Unternehmen oder Behörden gesucht, welche nicht aus der IT Industrie stammen, aber aktuell Open Source Software oder Codes auf Github oder im Internet unter einer Open Source Lizenz veröffentlicht haben oder planen dies zu tun. Als erstes würde mich interessieren, wie die Idee beim KAIO entstanden ist, eigene Projekte, Software oder Teile der Software zu veröffentlichen und der Allgemeinheit kostenlos zur Verfügung zu stellen?

B: Es waren vor allem vier Punkte, welche bei der Erarbeitung des Service im Vordergrund standen: Einerseits wollen wir mehr Wettbewerb unter unseren Lieferanten fördern und damit unsere Abhängigkeit zu den Lieferanten minimieren. Andererseits wollen wir mehr Transparenz schaffen und auch das Öffentlichkeitsprinzip fördern. Wir möchten von dem Umstand wegkommen, von einem Lieferanten über Jahre gebunden zu sein. Auch wenn eine Applikation von einem einzelnen Lieferanten entwickelt wurde, kann die Weiterentwicklung zukünftig von verschiedenen Firmen übernommen werden, sofern wir sie öffentlich zur Verfügung stellen.

Ein weiterer Aspekt ist die einfachere Kooperation zwischen Behörden und Kantonen. Grund dafür ist, dass alle Kantone ähnliche Aufgaben haben; jeder aber die Lösungen selbst realisiert. Nur in sehr wenigen Fällen findet heute eine interkantonale Zusammenarbeit statt. Wenn wir die Software veröffentlichen versprechen wir uns davon, dass die öffentlichen Verwaltungen gegenseitig voneinander profitieren und sich unterstützen können. Dies kann auf verschiedene Art und Weise geschehen: zum Beispiel in Form einer Community oder auch durch direkte Kontakte, welche sich dadurch ergeben können.

Ein weiterer Punkt ist die Wiederverwendbarkeit von Software. Wir haben uns bei der Erarbeitung darauf geeinigt, dass der Kanton nicht nur ganze Applikationen veröffentlichen soll, sondern auch einzelne Komponente daraus, sofern diese von öffentlichem Interesse sein könnten. Wenn man ganze Applikationen betrachtet zeigt sich, dass diese allerdings in vielen Fällen sehr proprietär sind. Einzelne Programmteile daraus kann man aber in vielen Fällen sehr wohl

anderen zu Verfügung stellen, welche dann davon profitieren können.

I: Welche Software ist zum Beispiel proprietär?

B: Als Beispiel kann ich ihnen unsere Steuersoftware nennen, welche für die Steuerveranlagung, Steuerberechnung und dem Inkasso eingesetzt wird.

I: Also das TaxMe?

B: TaxMe ist nur das Portal, um die Steuererklärung auszufüllen. Dies gibt es heute bereits in verschiedenen Kantonen in verschiedenen Ausprägungen. Nein, es geht hier um die nachgelagerte Verarbeitung der Daten in der Steuerverwaltung. Dazu setzt der Kanton eine Applikation ein, welche gemäss unseren Spezifikationen entwickelt wurde.

Eine solche Software ist sehr proprietär, weil sie die gesetzlichen Bestimmungen des Kantons Bern abbildet. Andere Kantone haben natürlich dieselben Aufgaben, jedoch andere gesetzliche Regelungen.

Wir haben geprüft, ob einzelne Komponenten veröffentlicht werden könnten. Im Laufe der Prüfung wurde jedoch festgestellt, dass es anderen Kantonen keinen Mehrwert bringt. Einerseits aufgrund der Spezifikation des Kanton Berns anderseits aber auch, weil die Software ursprünglich nicht für eine Veröffentlichung als Open Source Software entwickelt wurde. Es hätte einen erheblichen Aufwand für Anpassungen zum Beispiel der Dokumentation, Strukturen, etc. bedingt.

Der letzte der vier am Anfang erwähnten Punkte ist die Transparenz an und für sich und dass sich der Kanton nach aussen hin öffnen will.

I: Und von wem aus kam dies explizit?

B: Im grossen Rat wurden zwei Motionen eingereicht. Die Motionen haben gefordert, dass man das Thema Open Source im Kanton Bern fördern soll. Dazumal waren die rechtlichen Grundlagen im Kanton dafür jedoch noch nicht geklärt. In einem ersten Schritt musste deshalb zuerst geklärt werden, ob der Kanton Bern überhaupt Open Source publizieren darf.

I: Also kam dieser Wunsch nicht von den Mitarbeitern aus?

B: Nein, der Anstoss kam von Seiten der Politik.

I: Wenn man es mit einer Firma vergleichen würde, könnte man von einem Top-down Ansatz sprechen?

B: Ja, wenn man es so betrachtet, ist es ein Top-down Ansatz, denn der Grosse Rat ist unser oberstes Organ.

I: Danach habt ihr eine Klärungs-, Design-, Umsetzungsphase gehabt?

B: Genau. Die Abklärung von Professor Tomas Poledna und Professor Simon Schlauri hatte das Ziel, das Thema von der gesetzlichen Seite her zu untersuchen. Die beiden Herren haben dazu eine umfangreiche Studie erarbeitet.

I: Welche Schwierigkeiten hattet ihr von der Klärung bis zur Umsetzung?

B: Die Klärung fand, wie vorhin erwähnt, auf der juristischen Ebene statt. Dabei ist klar herausgekommen, dass der Kanton Bern ohne eine Gesetzesänderung Open Source publizieren darf. Die Ergebnisse wurden in einer 150-seitigen Studie dargelegt. Quintessenz aus der Studie ist, dass der Kanton keine Gesetzesänderung machen musste, damit man Software publizieren darf.

In der Designphase haben wir einen Business Case erstellt, in welchem dargelegt wurde, was die Ziele und der Inhalt der Tätigkeiten im Bereich Open Source für den Kanton Bern sind.

Die Evaluation der Lizenz und der Plattform war ein weiterer Punkt. Die Plattform auszuwählen war nicht so schwer, es gibt zwei bis drei weit verbreitete und etablierte Plattformen, welche zur Auswahl standen. Wir haben uns dann für GitHub entschieden.

Die Wahl der Lizenz war ein komplexeres Thema, welches wir in Zusammenarbeit mit unseren internen Juristinnen und Juristen geklärt haben. Die Frage war, welche Lizenz für den Kanton überhaupt geeignet war. Wir haben alle verfügbaren Lizenzen überprüft und analysiert. Schlussendlich ist die Wahl auf BSD gefallen. Im Anschluss daran haben wir die Prozesse und Checklisten erarbeitet. In dieser Phase haben wir mit der BVE eine Direktion involviert, welch mit dem Smart Auszug bereits Software publiziert hat.

I: Für was steht BVE?

B: Bau-, Verkehrs- und Energiedirektion. Dort ist auch das Thema Geodaten angesiedelt. Der Smart Auszug wurde durch die BVE bereits im Vorfeld unserer Arbeiten auf Bitbucket publiziert; somit konnten wir von ihren Erfahrungen profitieren.

Die BVE konnte uns sehr viele Inputs liefern und uns bei der Erarbeitung der Rahmendokumente und Checklisten unterstützen. Damit konnten wir auf Anhieb einen sehr guten Stand der Dokumente erreichen. In der Umsetzungsphase wurde dann die auf Bitbucket publizierte Applikation auf unsere GitHub Plattform migriert.

I: Was war das für eine Software?

B: Das ist eine Software, welche alle Angaben zu einem bestimmten Grundstück liefert. Sei es die Fläche, Grundbuchauszug, Altlasten, Belastungen oder Gewässerschutzsachen und so weiter. Als Privatperson kann man diese Angaben auf der ÖREB – Seite Kanton Bern abrufen. Es gibt auch vom Bund eine entsprechende Seite, das Handling ist jedoch eher kompliziert. Die BVE hat zusammen mit der Universität Bern einen Smart Auszug geschaffen. Dabei werden alle Daten auf eine einfache Art und Weise zusammengezogen und dem Benutzer in einem PDF zur Verfügung gestellt.

I: Habt ihr noch weitere Software, welche ihr veröffentlicht habt?

B: Nein, aktuell ist dies die einzige Applikation.

I: Wurde die Software von Anfang an als Open Source Software entwickelt worden?

B: Ja, das war von Beginn ab so geplant. Es war eine Zusammenarbeit zwischen dem Kanton Bern (BVE), der Uni Bern und zwei oder drei weiteren Kantonen. Die Software wurde durch die Firma Sturm und Bräm entwickelt. Wie gesagt war von Anfang an vorgesehen, die Applikation auf Bitbucket zu publizieren.

I: Wie ist man bei der Entwicklung vorangegangen? Also wann hat man es publiziert, am Anfang oder als fertige Lösung?

B: Dazu kann ich keine Aussagen machen, da die Entwicklung und Erstpublikation vor dem Beginn unserer Arbeiten erfolgt ist.

I: Sie haben vorher die gewissen Vorteile als eure Treiber genannt. Sehen Sie noch weitere Vorteile, wenn der Kanton Open Source Software veröffentlicht?

B: Ja, ein Aspekt ist natürlich auch der Kostenfaktor. Wenn zum Beispiel mehrere Kantone voneinander profitieren können, sollte das Ganze schlussendlich auch günstiger werden. Das heisst, dass bei einer Zusammenarbeit mit anderen die Kosten geteilt bzw. minimiert werden können.

I: Könnte es auch als Rekrutierungstool dienen?

B: Wie meinen Sie Rekrutierung?

I: Zum Beispiel um IT Spezialisten, welche viel Knowhow im Open Source Bereich haben, anzustellen.

B: Das Problem ist wie am Anfang des Interviews erwähnt, dass der Kanton selbst nichts entwickelt. Wir kaufen Entwicklungsleistungen extern ein. Wenn wir eine neue Software entwickeln lassen und veröffentlichen wollen, haben wir das Entwicklungs-Know-how nicht. Wir können daher keine Open Source Entwickler engagieren, da die Entwicklung extern erfolgt.

I: Welche strategische Bedeutung hat Open Source Software für KAIO?

B: Da wir am Anfang stehen, ist die strategische Bedeutung noch nicht so klar festgelegt. Aber ich könnte mir vorstellen, dass das Thema mit zunehmender Bedeutung und Verbreitung strategisch relevant werden könnte. Im Moment ist es jedoch freiwillig. Die strategische Bedeutung zeigt sich auch darin, dass es momentan noch freiwillig ist.

I: Und haben Sie eine Open Source Strategie?

B: Nein eine Strategie haben wir nicht.

I: Und ist sie irgendwie in die IT Strategie implementiert?

B: Neuerdings ist in unseren Richtlinien der Beschaffung erwähnt, dass wir wenn möglich auf Open Source setzen sollten. Aber es ist einfach eine Empfehlung.

I: Kommen wir zum Thema der Veröffentlichung. Sie haben gesagt, dass das KAIO nichts selbst entwickelt. Wie bringt da KAIO ihre Lieferanten dazu, die

Freigabe zu geben, damit die Ämter die Software veröffentlichen kann?

B: Bei Sturm und Bräm beim ÖREB Auszug war es von Anfang an klar. Aber wie es zukünftig aussieht, wird sich zeigen. Es wird jedoch zukünftig gegebenenfalls eine Anforderung sein. Wenn wir Ausschreibungen machen, wollen wir es unter Umständen als Bedingung fordern, dass wir es als Open Source publizieren dürfen.

I: Also entscheidet das KAIO eigentlich, dass es publiziert wird?

B: Zusammen mit dem Amt, welches die Software beschafft.

I: Und es kommen nur noch Lieferanten in Frage, welche dem entsprechen?

B: Die Lieferanten müssen es zukünftig quasi als ein Muss-Kriterium erfüllen. Wir wollen dies in den Ausschreibungsprozess integrieren, so dass es schon von Anfang an klar ist.

I: Was ist eurer Meinung nach, der beste Moment für eine Veröffentlichung?

B: Da habe ich noch zu wenig Erfahrung. Aber wenn es von Anfang an klar ist, dass die Software im Hinblick auf das entwickelt wird, denke ich, wenn sie einen gewissen Stand hat.

I: Gibt es auch eine Angst bezüglich Trittbrettfahrer oder besteht eine Imitationsgefahr?

B: Ja, das haben wir bemerkt. Wir hatten bei der Erarbeitung noch einen zweiten Player gehabt, die Steuerverwaltung. Wir wollten im Rahmen eines Pilots das NESKO Frontend veröffentlichen. Danach haben wir bemerkt, dass dies nicht geht. Anschliessend haben zwei Komponenten daraus evaluiert und soweit aufbereitet, dass die Steuerverwaltung entscheiden konnten, ob sie eine Publikation wollen oder nicht.

Schlussendlich sind sie zum Schluss gekommen, dass sie es nicht wollen. Denn sie haben gesagt, dass sie so viel Arbeit reingesteckt haben und sie sind, was die Steuerverarbeitung betrifft, schweizweit die Nummer zwei oder drei. Wenn sie die Applikation publizieren, besteht die Gefahr, dass jemand ihre Software nehmen, modifizieren und verbessern kann. Also diesen Aspekt gibt es und die Steuerverwaltung hat sich danach dagegen entschieden.

I: Welcher Teil des Codes wird veröffentlicht? Was ist euer Ziel? Den ganzen Code zu veröffentlichen oder nur einzelne Teile davon?

B: Es hängt ein bisschen von der Wiederverwendbarkeit ab. Wenn es Sinn macht, das ganze Programm als Programm zu veröffentlichen, dann wollen wir eigentlich das Ganze. Aber wenn es Sinn macht, nur gewisse Funktionen oder Funktionalitäten zu veröffentlichen, ist dies auch eine Möglichkeit. Es gibt zwei verschiedene Prozesse: einen für ganze Softwares und einen für die Libraries, womit beide Aspekte abgedeckt sind.

I: Wie viele Leute bei KAIO beschäftigen sich mit Open Source Software?

B: Null Komma zwei. Eigentlich bin es ich, der das Thema im Moment betreut und bewirtschaftet. Und es beschränkt sich eigentlich auf eine Initialberatung. Das ist das, was wir unseren Direktionen anbieten: Unseren Service vorstellen und aufzeigen worauf es ankommt.

I: Sie sind ein Berater des Kantons bezüglich Open Source Software?

B: Ja, das kann man so sagen. Obwohl vom Know-how her bin ich vielfach auf externe Unterstützung angewiesen, vor allem bei der Erarbeitung. Aber ich bin im Moment im KAIO die einzige Person, welche sich mit dem Thema beschäftigt.

I: Wie gehen sie auf alle Gemeinde des Kantons zu?

B: Die Gemeinden sind momentan nicht involviert. Im Moment beschränkt es sich auf die kantonale Verwaltung ohne Gemeinden. Unser Prozess ist so geregelt, wie auch bei anderen Themen, dass sich eine Direktion oder ein Amt bei uns meldet: „Ich will Open Source publizieren und wo könnt ihr uns unterstützen“. Die Idee ist, dass wir dann eine Initialberatung machen, unseren Service vorstellen, sie auf die Stolpersteine aufmerksam machen, aufzeigen was wichtig ist und worauf sie achten müssen. Bis jetzt hat dies leider noch niemand in Anspruch genommen. Danach ist die Idee, dass der Kunde das mit ihren Lieferanten selbst abwickelt. Und am Schluss melden sie uns, dass die Software publiziert wurde. Sie publizieren das selbst. Der Lieferant erhält einen Account und publiziert das danach selbständig.

I: Wenn ich es richtig verstanden habe: Das KAIO hat auf GitHub den ganzen Leitfaden publiziert und wie haben die Verwaltungen von dem erfahren?

B: Man hat intern über die gängigen Kommunikationswege informiert. Wir haben im Kanton verschiedene Fachgremien; darüber wurden sie informiert.

I: So wie ich es verstehe betreibt ihr kein aktives Push-Marketing, sondern wartet bis die Leute auf euch zukommen?

B: Genau. Die Idee war, wir publizieren es, schalten es frei, informieren und danach schauen wir, ob etwas passiert. Das war im letzten Herbst. Die Idee war auch, dass sich das eine oder andere Amt meldet, aber das war bisher nicht der Fall. Jetzt werden wir aktiv wieder informieren und in den verschiedenen Gremien Kurzpräsentationen halten. Damit wir es wirklich pushen können. Wir wollen in diesem Thema wirklich aktiv werden.

I: Also Präsentation habt ihr bisher noch nicht gehalten?

B: Nein, haben wir nicht.

I: Habt ihr eine offizielle Open Source Compliance oder Richtlinien.

B: Was meinen Sie genau?

I: Dass die Verwaltungen zum Beispiel wissen, was man darf und was nicht.

B: Das ist eigentlich rudimentär in diesen Dokumenten von unserem Service beschrieben. Es gibt Checklisten, in welchen klar geklärt werden muss, ob es irgendwelche Geheimnisverletzungen gibt, wenn wir die Software publizieren. Ich komme wieder auf die Steuerverwaltung zurück. Gewisse Algorithmen darf man zum Beispiel nicht publizieren, weil es sonst Rückschlüsse zulässt, wie die Steuern berechnet werden und solche Sachen. Das gibt es, ja.

I: Und habt ihr einen offiziellen Prozess für die Entwicklung, Veröffentlichung und Wartung von Open Source Software?

B: Das ist in diesen Dokumenten beschrieben. Das ist genau der Inhalt des Service. Was die Anforderungen sind, wie es entwickelt, dokumentiert werden muss. Das ist alles beschrieben.

I: Und wer entscheidet schlussendlich ob es publiziert werden darf?

B: Das Fachamt. Also das Amt, welches die Software entwickelt, ist auch für die Software verantwortlich. Das ist die sogenannte Fachverantwortung. Und sie entscheiden, sei es auf der Stufe des Amtes oder der Direktion, ob man das darf und publiziert es dann. Das liegt nicht in der Verantwortung des KAIO.

I: Wie würde man danach mit externen Open Source Anfragen umgehen?

B: Also, wenn jemand die Software verwenden will?

I: Genau.

B: Die steht ja frei zur Verfügung und ist für alle frei verfügbar.

I: Aber es könnte ja trotzdem irgendwelche Anfragen von externen geben.

B: Das wird durch das Fachamt bewirtschaftet. Wenn etwas käme, wenn sie es zum Beispiel nicht einfach herunterladen und verwenden können, dann geht es über das Fachamt.

I: Und sind immer Kontaktdaten vom Fachamt hinterlegt?

B: Ja, genau.

I: In der Checkliste gibt es einen Punkt, der lautet: Die Publikation von Open Source Software trägt dazu bei, dass finanzielle Risiken, Sicherheitsrisiken oder sonstige Risiken für den Kanton reduziert werden. Wie weist man dies nach?

B: Auf welcher Checkliste?

I: Das habe ich auf GitHub gesehen.

B: Können sie die Frage nochmals wiederholen?

I: Die Publikation von Open Source Software trägt dazu bei, dass die finanziellen Risiken, Sicherheitsrisiken oder sonst welche Risiken für den Kanton reduziert werden.

B: Das muss das Fachamt abklären. Aber da haben wir keine entsprechenden Hilfsmittel mit irgendwelchen Strukturen, mit der sie das

strukturiert machen können. Das müssen sie einfach abklären.

I: Habt Ihr auch Richtlinien, wie externe Software-Komponenten in die eigene Software integriert werden dürfen?

B: Ja hat es auch. Das ist auch in diesen Checklisten drin. Da geht es vor allem um die Dokumentation und die Erwähnung von Entwicklern. Ja das ist vorhanden.

I: Sie haben vorher die BSD Lizenz erwähnt. Unter welchen Lizenzen werden die Projekte veröffentlicht?

B: Unter BSD.

I: Alle?

B: Alle, ja.

I: Auf GitHub sind aber noch mehrere Lizenzen beschrieben.

B: Ja, aber eigentlich ist es die BSD Lizenz, die verwendet werden muss.

I: Wieso BSD?

B: Es geht dort vor allem um die Verantwortung und Bindung. Ich will mich aber nicht zu fest aus dem Fenster lehnen.

I: Das waren die Ergebnisse der von Ihnen erwähnten juristischen Abklärungen?

B: Ja, man kann das so zusammenfassen.

I: Und wie stellt das KAIO sicher, dass die Open Source Lizenzen eingehalten werden?

B: Dafür ist der Lieferant und das Fachamt verantwortlich.

I: Und wer kontrolliert dies?

B: Das Fachamt muss dies kontrollieren.

I: Hattet ihr schon Lizenzkonflikte gehabt?

B: Nein. Es wurde ja auch erst nur eine Software publiziert.

I: Und wie werden die Mitarbeiter von der Verwaltung für den Umgang mit Open Source Software geschult?

B: Gar nicht. Es gibt keine Schulung.

I: Ist so eine Schulung eventuell geplant?

B: Im Moment nicht.

I: Und wie könnte man eurer Meinung nach, das Lizenzrecht den Entwicklern oder den Mitarbeitern der Verwaltung spannend erklären?

B: Die Frage ist, ob das überhaupt nötig ist. Denn wir entwickeln ja nicht selbst. Während der Erarbeitung gab es ja eine relativ komplexe, juristische Abklärung, aufgrund derer wir schlussendlich diese Lizenz gewählt haben.

I: In der Open Source Welt sind Partner und Communities ziemlich entscheidend. Und wie motiviert das KAIO externe bei euren Open Source Software beizutragen?

B: Je nach Grösse und Art der Software haben wir im Sinne, dass die Direktionen und Ämter Communities bilden sollten. Dort wären sie dann relativ stark eingebunden. Aber sonst haben wir uns darüber nicht gross Gedanken gemacht. Wir haben uns gesagt, dass wir es zur Verfügung stellen und dann einmal schauen wie es sich entwickelt. Es sei denn, wir publizieren eine grosse Software. Wir haben GERES. Ich weiss nicht ob das Ihnen was sagt.

I: Nein.

B: Das ist das Gemeinderegister. Das haben 16 Kantone im Einsatz und dort gibt es zum Bespiel eine Community; die Applikation ist aber nicht Open Source. Wir haben GERES für uns als Beispiel genommen und uns gesagt, wenn wir eine grosse Software publizieren wollen, dann werden wir eine Community gründen. Bezüglich kleinen Applikationen haben wir uns nicht allzu grosse Gedanken über Communities gemacht. Also bei Libraries macht dies sowieso keinen grossen Sinn. Und bei grösseren Software Applikationen wie das ÖREB, dort gibt es eine Art Community. Ist aber nicht offiziell, sondern die Kantone, welche an der Entwicklung mitgearbeitet haben, treffen sich regelmässig; mehr oder weniger strukturiert. Dort hat man den Ansporn, dass man einander mitzieht und animiert weiterzuentwickeln.

I: Und wer nutzt momentan die Software, welche das KAIO veröffentlicht hat?

B: Also primär der Kanton Bern. Respektive eigentlich der Endbenutzer, also wir Bürger können diese verwenden. Aber für die Weiterentwicklung weiss ich jetzt nicht, ob

jemand anderes, ausser den Kantonen, die in der Entwicklung involviert waren, diese verwenden. Das wüsste ich nicht.

I: Also sind die Kantone Teil dieser Community?

B: Ja, aber nicht alle. Es sind, wie bereits gesagt, ungefähr drei oder vier Kantone, welche im Projekt mitgemacht haben.

I: Wie haben sie diese Kantone ausgewählt?

B: Das weiss ich nicht, das war vor meiner Zeit. Wir haben die fertige Software publiziert.

I: Private Entwickler sind jetzt nicht Teil der Community?

B: Nein. Der Hersteller Sturm und Bräm hat dort Einsitz in den Gremien, welche sich darum kümmern. Aber Private wüsste ich nicht, dass da jemand mitentwickelt.

I: Wie würde das KAIO eine eigene Community aufbauen? Ich habe gesehen, dass das KAIO einen Leitfaden für das Community Management habt.

B: Ja das ist so eine Matrix, in welcher man Fragen beantworten muss und am Schluss kommt eine Empfehlung für eine Communityform heraus. Aber es hängt davon ab, was man publiziert und was man damit erreichen will. Wie weit will man selbst die Kontrolle haben über das Ganze, wie weit gibt man es frei. Am Schluss gibt es verschiedene Formen von Communities, welche als Ergebnis aus dieser Matrix herauskommen. Und es ist halt generell schwierig zu sagen, wenn wir die Software und Anforderungen nicht vor uns haben. Das Fachamt muss im jeweiligen Fall definieren und entscheiden, was es will.

I: Die Entscheidung bezüglich zentraler oder dezentraler Kontrolle liegt also jeweils beim Fachamt?

B: Genau. Denn sie haben teilweise für ihre Software Roadmaps mit klaren Zielen. Und wenn man in einem solchen Fall alles herausgibt, wird es schwierig. Da ist mehr Kontrolle gefragt. Oder etwas bei dem man sagt: Ok das geben wir jetzt frei, entwickeln selbst weiter und schauen nicht auf die Anderen. Zusätzlich nehmen wir dann vielleicht auch nicht viel oder gar nichts zurück. Abhängig von solchen Entscheidungen gibt es eine andere Coummunityform.

I: Und bei der Software, welche das KAIO veröffentlicht hat. Welche Form habt ihr da gewählt, wenn man der Matrix folgen würde?

B: Das war alles schon etabliert gewesen. Diese Matrix wurde da nicht angewendet. Und es gibt auch keine offizielle Communtiy in Form eines Vereins oder so, sondern wie bereits erwähnt lose, periodische Zusammenkünfte, bei welchen sie sich treffen. Die involvierten Kantone inklusive der Entwicklerfirma.

I: Und wurde die Software seit der Publikation weiterentwickelt?

B: Ich glaube es hat seither keinen Release gegeben. Aber sie arbeiten daran, es wird immer weiterentwickelt. Eine neue Publikation hat es jedoch seit der Initialpublikation nicht gegeben.

I: Gemäss der Matrix kann man die Form der Community auswählen. Hat das KAIO auch Anweisungen, wie man aktiv ein Community Management betreiben sollte?

B: Nein.

I: Bezahlt ihr auch externe Entwickler, um die Software weiterzuentwickeln?

B: Also im Fall ÖREB, ist eine externe Software Firma involviert und die zahlt man natürlich für ihre Entwicklungen. Aber sie hat den Lead.

I: Habt auch eine Community Governance etabliert?

B: Nein, da sind wir noch am Anfang.

I: Von der Eclipse Fundation gibt es ein Maturitätsmodell. Wo würdet ihr euch als Kanton da einschätzen?

B: Finde das jetzt noch schwierig zu beurteilen. (...) In praktisch all unserer Software, die entwickelt wird, haben wir eigentlich Fremdsoftware drin. Könnten Sie mir das Maturitätsmodel ein wenig erläutern?

I: Das No, wäre eine totale Ablehnung gegenüber Open Source, weil zum Beispiel der Support fehlt oder es nicht wettbewerbsfähig ist. Use bedeutet, dass man es nutzt.

B: Das machen wir sicher.

I: Contribute ist, dass man es nicht nur nutzt, sondern auch etwas zurückgibt. Und Champion wäre, dass man auch aktives Community Management betreibt und fördert. Und Investment bedeutet, dass man wirklich auch investiert. Bei Unternehmen wäre es zum Beispiel eine Community sponsert oder Mitarbeiter für das Community Management einstellt.

B: Also in all unseren Softwareteilen sind Open Source Komponenten enthalten. Das Use trifft sicher zu. Beim Contribute sind wir am Anfang. Schwerpunktmässig sind wir beim eins und möchten uns Richtung zwei bewegen, das wäre unser Ziel.

I: So wie ich es aus dem bisherigen Interview verstanden habe, hat das KAIO als Steuerungsorgan diese Vision, jedoch ist die Kontrolle bei der Fachabteilung.

B: Ja das ist korrekt.

I: Einzelne Mitarbeiter bei der Verwaltung haben ja einen definierten Arbeitsauftrag. Was das KAIO verlangt, bedeutet für jemanden, der es umsetzen sollte, schlussendlich eine Mehrarbeit.

B: Genau.

I: Wie bringt das KAIO diese Leute dazu, diese Mehrarbeit leisten? Sie werden ja nicht von anderen Aufgaben befreit, wenn sie dies machen, oder?

B: Gut, primär ist es in unserem Fall ein finanzieller Mehraufwand, denn wir machen es ja nicht selbst. Man muss die Finanzverantwortlichen dazu bringen, mehr Ausgaben dafür zu tätigen. Dort ist es auch noch zweigleisig. Einerseits haben wir festgestellt: wenn wir bestehende Software publizieren wollen, müssen wir sie zuerst anpassen. Wie im Fall der Steuerverwaltung. Dort hätte es ziemlich viel gekostet. Was wir uns aber vor allem versprechen ist, wenn wir Neuentwicklungen machen und eine Publikation von Anfang an im Fokus haben, dass der finanzielle Mehraufwand viel kleiner ist. Und ich denke aus diesen Erfahrungen, welche wir gemacht haben, müssen wir die Entscheidungsträger motivieren, diesen kleinen Teil an Mehraufwand bei Neuentwicklungen zu tragen. Da sind wir überzeugt, dass wir das eher können, als bei einer bestehende Software Geld nur für die Publikation zu investieren. Also motivieren müssen wir vor allem die Leute, welche über Geld entscheiden und nicht über eine Mehrarbeit in dem Sinne.

I: Man sagt, dass private Entwickler in der Open Source Welt entweder intrinsisch oder extrinsisch motiviert sind. Die intrinsischen haben Spass am coden und finden es eine gute Sache und die extrinsische coden, weil sie entweder bezahlt werden oder sich zum Beispiel einen Imagegewinn davon versprechen. Wenn man dieses Konzept auf die Verwaltung herunterbricht. Wie könnte man sie intrinsisch motivieren?

B: Das wird schwierig (lacht). Also rein durch Motivation, das haben wir jetzt auch bemerkt, hat niemand das Gefühl, es zu wollen, weil es gut ist. Ich glaube wir haben ihnen etwas Gutes zur Verfügung gestellt. Ich habe das Gefühl, wir müssen nicht befehlen, weil das können wir nicht. Aber ich glaube, wir können versuchen, es extrinsisch zu treiben. Dass wir unseren Kunden sagen, dass es nun ein Teil davon ist und Sinn macht. Intrinsische Motivation im Kanton Bern ist meiner Meinung nach, wenn es sowieso noch etwas kostet und einen Mehraufwand gibt, eher nicht zielführend.

I: Wie ich das verstanden habe, tragen die Fachämter die Kosten für die Publikation von bestehender Software selbst oder gibt es einen gemeinsamen Topf dafür?

B: Nein es gibt keinen Topf.

I: Würde so etwas helfen?

B: Das würde unter Umständen vielleicht helfen. Aber das Problem ist, dass wir unsere finanziellen Kompetenzen an die Ämter delegiert haben. Das Budget ist dezentral verteilt. Jede Direktion hat sein eigenes Budget. Wir haben nahezu kein zentrales Budget mehr. Also sie sind selbst für das Geld verantwortlich, bezüglich Budgetierung, was sie machen und was sie ausgeben. So ein Topf würde dieser Strategie dann eigentlich widersprechen. Aber es könnte motivieren, wenn sie es nicht selbst bezahlen müssten.

I: Spannend. Wenn ich es richtig verstanden habe, versucht das KAIO bei Neuentwicklungen, Open Source im Ausschreibungsverfahren so zu verankern, dass zukünftig mehr Projekte publiziert werden könnten.

B: Ja das ist die Idee.

I: Ich habe noch zwei, drei Frage über die Publikation. Ihr habt eine Checkliste für die

Publikation. Wie werden Software publiziert? Welche Prozesse muss man dafür durchlaufen?

B: In der Publikation selbst sind wir nicht involviert. Das sind Checklisten, in welchen die Kunden die verschiedenen Punkte erfüllen müssen; wenn es einen Stopper hat, dürfen sie es nicht publizieren. Entscheiden, ob es publikationsfähig ist und es physisch publizieren, machen sie alles selbst.

I: Wird alles zentralisiert über ihre GitHub Plattform publiziert oder haben die Verwaltungen auch eigene GitHub Accounts?

B: Nein es wird alles über die Plattform, welche wir jetzt erstellt haben, publiziert. Und sie erhalten einfach Zugriff auf die Plattform.

I: Sie haben vorher erwähnt, dass das KAIO keine Schulungen hat.

B: Ja.

I: Könnte das vielleicht auch ein Instrument sein, andere Verwaltungen zu motivieren, da mehr zu machen.

B: Die Frage ist was Sie unter Schulung verstehen. Wenn es darum geht, unseren Prozess und unsere Hilfsmittel zu erläutern, dann haben wir das und wollen wir das. Das ist unsere Beratungsteil, welchen wir möchten. Aber wenn es darum geht, jemanden im Thema Open Source selbst zu schulen, dann haben wir nichts vorgesehen und ich denke es macht bedingt Sinn, weil wir ja nicht selber publizieren.

I: Also im Fall von KAIO sehen Sie die Schulung eher in Form eines Workshops?

B: Genau das ist auch die Idee.

I: Bei wem halten sie ihre Präsentationen?

B: Einerseits wird sicher ein Teil der Geschäftsleitung des Amtes oder der Direktionen dabei sein, aber natürlich auch der IT Bereich sowie Projektleitende und Fachverantwortliche für Applikation. Einfach diejenigen, welche später dann auch die Entscheidungsträger sind.

I: Wäre es auch eine Idee die Mitarbeiter der operativen Ebene ins Boot zu holen?

B: Ja das ist sicher vorstellbar. Man kann das auch mehrmals machen. Da haben wir einfach noch keine Erfahrungen. Denn bis jetzt ist noch niemand auf uns zugekommen. Die Idee ist, dass wir unseren Service zuerst Top-Down auf der oberen Stufe vorstellen und danach weiter unten. Das machen wir auf jeden Fall.

I: Die Führungskräfte zuerst, weil sie entscheiden und das Budget verwalten, oder?

B: Ja genau. Also wir stellen sowohl die finanziellen wie auch die fachlichen Aspekte vor. Damit man die Entscheidungsträger auf Themen wie zum Beispiel heikle Algorithmen welche Rückschlüsse zulassen sensibilisiert oder dass die Applikation ohne Daten in der Software publiziert und solche Themen.

I: Wie viele Verwaltungen und Fachämter gibt es im Kanton Bern?

B: Fachämter weiss ich nicht genau. Aber es gibt sieben Direktionen. Das sind schätzungsweise zwischen 50 und 60 Fachämter. Nur ein Teil davon befasst sich mit solchen Themen.

I: Vorher haben wir von Vorteilen gesprochen. Sehen sie auch Risiken in der Veröffentlichung?

B: Also das ist eigentlich das konträre zu den Vorteilen. Bei der Erarbeitung ist herausgekommen, dass man zum Beispiel Wettbewerbsverluste haben kann, wenn man Sachen offenlegt, das war ein Thema.

I: Wettbewerbsverlust für wen?

B: Zum Beispiel für eine Steuerverwaltung, welche ihre Software publiziert, dann im Ranking, welches es in der Schweiz gibt, ihren Platz verliert und ein anderer Kanton davon profitieren könnte. Zu diesem Thema haben wir noch nicht so viel Erfahrungen.

I: Im Bund gab es einen Artikel darüber, dass der Kanton Bern ein Wettbewerber gegenüber anderen IT Firmen sei, weil sie Software veröffentlichen. Was sagen sie zu dem?

B: Das war eines der Hauptthemen, welches in der vorgelagerten Studie geklärt werden musst. Man ist zu Schluss gekommen, dass dies in den meisten Fällen nicht der Fall ist, weil wir sehr proprietäre Software entwickeln.

I: Sie haben vorher erwähnt, dass es ein Ziel ist, dass es günstiger wird. Gleichzeitig haben Sie vorher auch erwähnt, dass die Fachämter Geld in die Hand nehmen müssten.

B: Also Geld in die Hand nehmen müssen sie vor allem bei der Portierung von bestehender Software, dass wirklich alles den Anforderungen von Open Source entspricht. Und vielleicht auch ein bisschen mehr, wenn man neu entwickelt. Aber was wir uns davon versprechen ist, dass man dann zusammen weiterentwickeln kann, dabei Kosten sparen und die Kosten verteilen könnte.

I: Wie wollen Sie die Fachämter überzeugen, dass vielleicht kurzfristig die Kosten steigen und langfristig es etwas bringt?

B: Sagen wir es mal so. Wenn man es einfach publiziert, ohne eine Community zu haben, dann wird es sehr schwierig. Ich denke im Rahmen einer Community, kann man ihnen schon näherbringen, dass man davon profitieren könnte. Aber wenn man es einfach publiziert und denkt, ja vielleicht macht ja jemand etwas, dann wird es schwierig. Dann wissen wir nicht, ob wir profitieren könnten.

I: Ist der Communityaufbau oder Beitritt bei bestehenden Communities, Teil ihrer Beratung?

B: Der Aufbau ist im Sinne der Matrix, in welcher man schlussendlich zu einer geeigneten Struktur kommt, ist Teil der Beratung. Es gibt natürlich auch verschiedene Rechtsformen von solchen Communities, wie lose Communities oder Vereine. In diesem Bereich haben wir uns mal ganz grob ein paar Gedanken gemacht. Aber konkret warten wir mal ab, bis es auftritt und jemand das Bedürfnis hat.

Interview mit David Ösch am 24.01.2019

I: Für die Einleitung habe ich ein paar persönliche Fragen. Was ist ihre Ausbildung, wie seid ihr zu Swisstopo gekommen und was ist ihre momentane Funktion?

B: Ich habe Geografie studiert und 2002 in phil nat promoviert. Ich bin dann in die Privatwirtschaft. In der Agrarindustrie bei der Syngenta, das ist ein globaler Multi. Ich habe mich dort zuerst einmal mit Cloudcomputing in Kombination mit Geoinformationsmanagement auseinandergesetzt. Dann bin ich zu Swisstopo gekommen und bin Projektleiter seit zehn Jahren für geo.admin.ch Geoportal Bund.

I: Habe zuerst noch eine Frage bezüglich geo.admin.ch und Swisstopo. Wie hängt das zusammen?

B: Am besten sieht man das auf geo.admin.ch/imperssum. Geo.admin ist das Geoportal des Bundes, der ganzen Bundesverwaltung. Wird von Swisstopo betrieben im Auftrag interdepartementale Koordinationsstelle GKG. Die Swisstopo ist ein Leistungserbringer für Geoinformationsdienste, also ist ein Betreiber wie zum Beispiel eine Swisscom, welche Telefoniedienste anbietet. Swisstopo hat aber auch selbst Daten in diesem Portal drin. Man hat also zwei Hüte an. Einerseits ist man Datenherr, zum Beispiel die Hintergrunddaten wie Landeskarten, mehr als die Hälfte der Daten werden von anderen Bundesämtern bereitgestellt, wie zum Beispiel des Bundesamtes für Umwelt etc. Betrieben durch KOGIS, das ist die Koordinationsstelle Geoinformation, ist auch bei Swisstopo angesiedelt. Wir sind Service Provider für andere Ämter. Das Ziel von geo.admin.ch ist die Umsetzung des Geoinformationsgesetz (GeoIG). Da ist auch das fördern von Open Data / Open Access Zugang zu Geodaten im Auftrag enthalten.

I: Wie ist der Unterschied zwischen Bundesamt für Landestopografie und Swisstopo?

B: Das ist das Gleiche Entität.

I: Das ist also quasi eine Brandname?

B: Ja. BAFU ist das Kürzel für Bundesamt für Umwelt und Bundesamt für Landestopografie, referenziert sich als swisstopo. Es gibt ja zum Beispiel auch Meteo Swiss und das wäre dann Bundesamt für Meteorologie und Klimatologie.

I: Kommen wir zur Idee. Wie ist die Idee bei Swisstopo entstanden, eigene Projekte, Software oder Teile der Software zu veröffentlichen und der Allgemeinheit kostenlos zur Verfügung zu stellen?

B: Ich schicke dir noch ein Dokument, wo es eigentlich noch gut beschrieben ist. Man hat das Geoinformationsgesetz, welches aussagt, dass der Bund einfach, schnell, kostengünstig Zugang zu Geoinformationen bereitstellen muss (Zweckartikel). Die Krux war, dass man Zugriff zu den Informationen oder zu den Daten bereitstellt, aber die Werkzeuge dazu, wie SWTools, um diese einzulesen oder darzustellen, standen den Nutzern ja nicht offen zur Verfügung. Das war ein Grund. Wir haben uns dann gesagt, wenn wir Prozessketten entwickeln, ein Portal entwickeln, dann sollten es andere nutzen können. Und das andere war, als wir es 2008 angefangen haben, gab es nichts COTS, also Commercial off-the-shelf was wirklich performantes Webviewiung von Geodaten zugelassen hätte. Es gab kein Framework, welches das map.geo.admin.ch wie wir es heute kennen, mit Slippy Maps wie bei Google Maps, schnell und einfach zugänglich macht. Das war der zweite Grund. Entweder wir beschaffen es und beim Bund dauern Beschaffungen sehr lange. Wir haben gesehen, dass es eine Community basiertes Open Source Framework gibt, welches schon mal 80 Prozent von dem kann, was wir für die Umsetzung eines Viewers benötigen. Wir hatten dann auch i interne Entwickler gehabt, welche sich da eingearbeitet und contributet haben. Dann haben wir das Portal aufgeschalten. Also es gab nichts Commercial off-the-shelf und zweitens wollten wir den Leuten nicht nur Zugang zu den Daten geben, sondern auch das Werkzeug wie sie die Daten visualisieren können. Was wir dann schön sehen ist, dass wenn man https://github.com/geoadmin/mf-geoadmin3 betrachtet, hat man unten im Readme andere Länder/Gov einheiten, welche dies kopiert haben. Also Luxemburg, Norwegen, Trentino, Piemont oder Thurgau haben den Source Code übernommen, eigene Portale aufgebaut mit gleichen Stacks und haben dafür wenig Umsetzungszeit benötigt und nicht ein Jahr für Eigenentwicklung / Beschaffung aufwänden müssen. Und sie haben den Code dann auch wieder in die Community reingespielt. Also

Sharing is Caring. Auch time to market war wichtig, denn wir haben gesehen, dass wir innerhalb von drei bis vier Monate etwas publizieren können, anstatt das sehr langen Beschaffungsverfahren durchzulaufen. Und wir sind unabhängig und es ist offen.

I: Unabhängig bezüglich?

B: Wir haben keinen Vendor-Lock-in und wir können den Code analysieren. Der technische Vorteil ist, dass wir eine Schweizer Projektion haben, wo wir sicherstellen müssen, wie die Daten dargestellt werden. Die Projektion, welche wir haben, wirklich auch stimmen. Wir müssen zum Beispiel den Source Code anschauen können, dass es genau um so und so viel Kommastellen berechnet wird. Das können wir so sicherstellen.

I: Haben sie vom Bund dafür eine Genehmigung gebraucht?

B: Wir haben am Anfang eigentlich selbstviel SW entwickelt, denn damals war es ganz neu und wir haben die Software benötigt. Dann war es ein Share-alike, also wenn wir Änderungen gemacht haben, mussten wir es wieder publizieren. Es war ein KOGIS Beschluss, dass wenn wir etwas machen, dann gibt man es auch der community zurück. Wir hatten eine SVN Versions Repository am Anfang, jetzt ist es auf Github. Jetzt ist es eigentlich Teil von der Codebase oder von der Lizenz, welche wir einsetzten, sagt eindeutig, dass man Änderungen veröffentlichen muss. Als Bundesstelle hat man entschieden, dass man mit Open Source Software arbeitet und damit akzeptiert man auch die share-alike Lizenzbedingungen. Jetzt haben wir eine BSD Lizenz drin. Es ist eine laufende Diskussion aud Ebene Bundesverwaltung im Gange bezüglich OSS. Also Matthias ist in dieser Arbeitsgruppe drin vom E-Gouvernement, von ISB, zur Ausarbeitung von Richtlinien, wie der Bund mit Open Source umgehen soll. Da gibt es zwei unterschiedliche Interpretationen von verschieden Exponenten, ob der Bund das darf oder nicht. Aber da kann dir Matthias die Unterlagen geben, die hat er.

I: War die ganze Idee eher Top-down oder Bottom-up?

B: Bottom-up. War ganz klar Bottom-up. Intrinsisch motiviert. Die Programmierer, welche auch gesagt haben, dass sie viel mehr Freude haben an etwas zu arbeiten, welches sie selbst beeinflussen und die Performance selber steuern können. Wir machen das Ganze via Cloudcomputing Ressourcen, was auch ein Grund war. Schon 2008 haben wir mit CC angefangen, waren einer der Ersten. Und auf der Cloud kann man nicht einfach Dongles vergeben, um zu lizenzieren. Da war es ganz klar, dass wir mit Open Source arbeiten müssen, wenn wir es horizontal skalieren wollen. Das war auch ein Grund, aber Bottom-up war der Treiber.

I: Was für Projekte haben sie jetzt unter einer Open Source Lizenz entwickelt?

B: Wir haben drei Schichten. Also wir haben eine Infrastrukturebene, eine Dienstebene und Viewer Ebene also Frontend. Auf der Infrastrukturebene entwickeln wir selbst weniger, sondern nutzen es vielmehr, also Linux-latest Debian Stretch. Wir nutzen Engines, zb Varnish oder Proxys. Es sind alles Open Source Komponente, welche wir nutzen. Unsere ganze Infrastruktur basiert rein auf Open Source, also Proxies etc. Auf dem ESB, der Enterprise Service Bus, haben wir OSS Datenbanken und OSS basierte Dienste (mf-chsdi3), welche wir nutzen. UMM Mapserver, das sind Lösungen, welche Geodienst und Geoservices bereitstellen, also Geoinformationsdienste. Und dort kam es auch vor, dass wir auf dieser Ebene in Open Source investiert haben. Wir comitten Bugfixing, Feedback geben oder Verbesserungsvorschläge implementieren. Das dritte Paket wo wir haben ist das Frontend, das ist das mf-geoadmin3. Auf Openlayers, AngularJS oder früher halt auch noch Bootstrap. Javascriptbibliotheken, welche wir angereichert haben und eine Nutzerinterface hat. Da haben wir sehr viel aus der Sicht der Open Source Community investiert. Aber das ist nur das Frontend. Die grössten Aufwände hatten wir im Bereich Infrastruktur respektive der Servicebereitstellung. Das sind die drei Ebene, in welchen wir investiert haben.

I: Und das steht alles im Zusammenhang mit diesem Maps Viewer?

B: Ja es steht alles mit diesen Geodienste im Zusammenhang. Das Ziel ist nicht, dass wir ein Viewer bereitstellen, sondern dass die Leute schnell, einfach und kostengünstig Zugang auf Geodaten haben. Und der Viewer ist so im Sinne wie "Eat our own Food". Wir stellen Geodaten zur Verfügung, mit denen man suchen, zeichnen, drehen oder überlagern kann. Wir zeigen eigentlich was man mit den Daten machen kann und da hat man auch das Framework dazu. Darum gibt es auch so viele Kopien von dem Portal, welches wir haben.

Zum Beispiel Schweiz Mobile, ein grosses Wanderportal. Das hat unsere Lösung genommen, kopiert und macht jetzt darauf Freizeitanwendungen. Fokus ist auf dem Dienst. Und Open Source Komponente, welches sich am schnellsten ändert ist das Frontend. Je tiefer man herunter geht, desto grösser ist release cycle. Stretch Debian hat seit zwei bis drei Jahren ein LTS Release. Frontend wechselt fast jedes Halbjahr bis ein Jahr. Und dort ist man mit Open Source sehr schnell vorne dabei, auf neue Geräte oder Touch Devices zu reagieren etc.

I: Kommen wir auf das Thema Motivation zu sprechen. Bei der Idee habt ihr schon gewisse Motivationen, wie schnellerer Beschaffungswege oder Freude der Mitarbeiter, genannt. Sehen sie weitere Chancen oder Motivationen, welche Swisstopo hat, die Projekte unter einer Open Source Lizenz zu veröffentlichen?

B: Man kann natürlich ganz klar auch die Leute viel besser sensibilisieren.

I: Interne oder externe Leute?

B: Extern. Wenn wir die Leute in der Nutzung der Geodaten schulen. Es ist offen. Man kann Werkzeug nicht kaufen, sondern man kann ein Werkzeug einfach nutzen. QGIS, das sagt ihnen sicher etwas, das ist Desktop GIS, was auch Open Source ist. Das ermöglichte, dass wir nicht nur offene Daten haben, sondern offene Schnittstellen auch propagieren. Das war eine wichtige Motivation, mit Open Source Software können wir auch sehr schnell auf offene Schnittstellen einsetzten und offene Schnittstellen fördern, was eine grössere Verbreitung und damit Nutzung der Daten mit sich bringt. Wir haben sicher auch plötzlich einen weltweiten Reach gehabt. Mit den Entwicklern weltweit, mit denen wir in Kontakt stehen. Man kann die Idee von Open Daten, Open Geodaten weltweit einbringen. Open Source Community, das sind Leute, die Open Source Software brauchen und die bedienen dann andere wiederum. Das heisst sie nutzen nicht nur Software, sondern nutzen auch unsere Daten damit. Plötzlich sind unsere Daten auch in andere Anwendungen zu finden, also es ist wie ein Multiplikator gewesen. Community ist wichtig, man schafft persönliche Kontakte. Die Schweiz ist relativ klein und ich glaube es ist ein grosser Vorteil, welchen wir haben. Die Schweiz ist wirklich klein, wenn wir Open Source Software Community betrachten. Hat auch dazu geführt, dass gewisse Kantone sich zusammengetan haben und GeoMapFish/geo gemacht haben. GeoMapFish ist ursprünglich eine Fork unserer Lösung, welche sie noch auf die Kantonsbedürfnisse angepasst haben, mit Identity, Accessmanagement, geschlossener Bereich etc. Und wenn wir einen Wunsch haben, etwas zu entwickeln, dann können wir es in die Runde werfen. Und vielleicht hat es jemanden anderes schon entwickelt, braucht es schneller und wir können es wieder nachher zurück bei uns einfügen. Also wenn ich Commercial-of-the-shelf nutze, dann kann ich auch eine Idee einbringen, aber dann muss ich häufig auf den nächsten Release warten. Dieser Release ist nicht nur von uns abhängig. Sondern irgendjemand kann einen Fork oder einen Branch machen, zum Beispiel ein ganz konkretes Beispiel Importwerkzeug im Viewer. Ich kann eigene GPS-Daten importieren über ein Modul, welches jemand anderes entwickelt hat. Wir haben es bei uns wiederverwendet. Dieser sehr kurze Releasecycle war wichtig. Ich persönlich finde wichtig, dass es sich Entwickler individuell einbringen können. Auch kleine Anpassungen von einem Entwickler kann man reinnehmen. Das schafft Identitär und Inklusion. Wenn jemand eine gute Idee hat, kann man sie schnell reinnehmen, wenn sie gut ist. Man muss nicht durch ein komplexes Konstrukt wie bei einer kommerziellen Software. Hat natürlich auch Nachteile, man kann sich natürlich auch verschiessen.

I: Wo sehen sie sonst noch Nachteile?

B: Nachteile ist immer bei Communities, dass die Leute kommen und gehen. Etwas ist heiss und dann ist es plötzlich nicht mehr heiss. Man muss aufpassen, dass man keine falschen Frameworks aufschwätzen lässt, sonst wird es sehr schnell teuer. Man muss die Community zusammenhalten. Man darf nicht den Weg einschlagen, wie wenn man es bei einer kommerziellen Lösung machen würde, dass man sich zu fest auf einen Pfad oder Produkt Openlayer stiert. Sondern muss auch zum Beispiel ein bisschen offen für andere Sachen sein. Und sonst ist man nachher genau dort, dass man sich seine kommerzielle Lösung aufbauen will. Es ist kein Nachteil, es ist halt eine Wahrheit: Open Source Software ist nicht billiger und dass meinen viele Stakeholder. Open Source Software ist am Ende für mich als Projektleiter genau gleich teuer wie kommerzielle Software. Ich investiere das Geld halt anders. Ich muss in die Schulung von Leuten investieren (was Nachhaltiger ist als in einen globalen Multi) und andere Risiken eingehen, welche auch Kosten. Plötzlich ist der Hauptentwickler weg und muss schauen wie

gross ist die Community etc. Ich muss auch schauen, dass es weitergeht. Aktuell machen wir ein Crowdfunding für Openlayers 6. Wir haben schon 200 000 USD in drei Woche zusammengekriegt, was nicht schlecht ist. Dann muss man diese Stakeholder akquirieren und halt alle einmal anschreiben. Es ist im Interesse des Bundes vorwärts zu machen und neue Komponente zu erhalten. Es benötigen zwar alle, jedoch hat niemand gesagt, dass sie es jetzt und heute wollen. Aber alle möchten es gerne. Es ist lustig, sobald man die Leute anschreibt und mitteilt, dass man einen Betrag investiert und wenn sie auch investieren, dann macht man es zusammen. Plötzlich fliesst das Geld und jetzt sind wir bei 200 000 Franken in drei Wochen.

I: Und wer sponsert da?

B: Es sind andere Kantone, also andere GOV Einheiten. Es sind aber auch Private oder grosse Multinationale Firmen, also wirklich grosse Firmen. Der Betrag variiert zum Beispiel von 500 Franken von Privatpersonen, 1000 Franken von einer Stadt bis zu 10000 Franken von einem Kanton oder bis zu 100000 von einer Organisation. Diese investieren und wollen diese Feature wirklich selbst für die nächsten zwei Jahren drin haben. Da ist aber sehr viel Zwischenmenschliches. Man muss sehr viel die Leute überzeugen. Und das schöne ist, dass die Leute forciert sind, um zu überlegen ob sie es machen wollen oder nicht. Das Problem, welche wir noch sehen ist, dass die Open Source Community nicht in allen Ländern die gleiche legale Entität ist. Es ist eine Community, kein legaler Körper und das ist auch eine Herausforderung für den Bund. Wir bei Swisstopo machen relativ viel Forschung, haben eine Forschungs- und Entwicklungsabteilung, also passt das dort rein. Das Problem ist einfach wie man die Contributions Requirements legalmässig verbindlich machen kann. Und das ist eben dort wo der Matthias beim ISB die Studie gemacht hat. Da musst du mal dort nachschauen, was da so der Stand ist.

I: Welche strategische Bedeutung hat Open Source Software für Swisstopo? Habt ihr eine offizielle Open Source Strategie?

B: Nein das haben wir nicht. Das ist wirklich Bottom-up getrieben. Es ist resultatorientiert, wenn wir für eine Ausschreibung für ein Projekt machen. Open Source ist für uns kein Ziel, sondern ist für uns eine Methode. Wir haben die Heilige Dreifaltigkeit, das sind Open Access, Offener Zugang zu Daten, Open Standard und Open Source. Wir denken, dass wir durch diese Kombination, das Ziel vom Geo IG erreichen. Denn mit offenem Zugang, offener Standard und das offene Werkzeuge dazu, können wir sicherstellen, dass die die Geodaten so schnell wie möglich vorhanden sind.

I: Was ist das Geo IG?

B: Geo Informationsgesetz. Das hat wirklich noch einen coolen Zweckartikel. Der Zweckartikel ist ziemlich legendär. Es gibt wenige Gesetze, welches so radikal in Zweckartikel ist. Im Sinne, welches sagt, dass alle Geodaten allen, also Bund, Kanton und der Bevölkerung für eine breite Nutzung, nachhaltig, rasch, einfach, in der erforderten Qualität und zu angemessenen Kosten, zur Verfügung stehen. Wenn du die Cloudcomputing Definition mit dem Geo IG vergleichts, dann ist das ungefähr gleich. Cloudcomputing sollte auch rasch, einfach und ontime, also auf die Sekunden genau Computing Ressourcen bereitstellen. Open Source wäre bei uns nie so gut angekommen, wenn wir Cloudcomputing nicht gehabt hätten. Cloudcomputing war ein Traum, weil man 1000 Server haben kann, die gleiche Software installieren und muss nichts extra 1000 Lizenzen zahlen. Das ist der Dongle, welche man nicht gebraucht hat.

I: Sprechen wir mal über die interne Organisation. Wie viele Leute bei Swisstopo beschäftigen sich mit Open Source?

B: Also Entwickler meinst du, oder?

I: Ja

B: Entwickler sind es ca. zwei in der Infrastruktur, ca. fünf auf der Serviceebene mit den Diensten, Backendservice sind es ca. drei und Frontend auch nochmals ca. drei. Man kann sagen, dass ungefähr sicher bei zehn Leute die tägliche Arbeit mit Open Source Software dominiert ist.

I: Und haben sie fixe Rollen in diesem Team?

B: Inwiefern meinst du?

I: Also zum Beispiel ein Entwicklungsteam, jemand der die Strategie vorgibt oder jemand der es überprüft.

B: Wir haben Entwickler, Projektleiter und Betreiber. Wir haben eigentlich ein DevOps

Team mit continious Integration. Wenn du DevOps hast, dann hast du die Rollen eigentlich auch drin. Plus Projektleitung, welche dann von aussen sagt, dass man das jetzt mit Open Source machen kann.

I: Und alle arbeiten jetzt zu 100 Prozent in diesen Open Source Projekte?

B: Ja.

I: Dürfen ihre Mitarbeiter auch an anderen Open Source Communities während der Arbeitszeit mitwirken? Wie viel Freiheit haben sie?

B: (...) Nein sie tragen während der Arbeitszeit nur auf/ Wir haben Openlayers Community wo wir arbeiten, das Mapping Framework. Aber wir haben auch noch PostGIS Community (enn wir einen Patches schreiben oder ein Feedback geben, ja. In der Freizeit arbeiten viele noch selbst auf privater Basis in anderen Communities mit. Aber auf Arbeitszeit nur was für uns relevant ist.

I: Haben sie offizielle Open Source Compliance und Richtlinien?

B: Wir halten uns an die Beschaffungsrechtliche Compliance. Also wenn wir ein Werk abnehmen oder ein Werk für Open Source in Auftrag geben. Dann nutzen wir die AGB des Bundes, dort ist auch Softwarebeschaffung drin. Allgemeine Geschäftsbedingung des Bundes, die kannst du herunterladen. Dort ist auch enthalten, wie man Open Source beschafft und wie man sie in den Betrieb übernimmt. Das ist unsere Compliance eigentlich. Wir haben eine Review Prozess. Swisstopo ist ISO zertifiziert, also wir haben einen Review. Es ist ein MS-Prozess wie wir Software entwickeln und abnehmen. Aber es hat vielmehr mit Softwareentwicklung zu tun, ob es jetzt Open ist oder nicht, spielt keine Rolle. Explizit Richtlinie für Open Source steht in der AGB des Bundes.

I: Wie sieht dieser Review Prozess aus?

B: Bei uns intern?

I: Ja

B: Wir schauen auf GitHub mit pull request und Vier-Augen-Prinzip. Dann haben wir einen Vorgesetzten, welcher es merged. Ist ein rein technischer Prozess, wo er nach dem ISO Prozess durchgehen muss.

I: Benutzt ihr auch Source Code Scanning Tools?

B: Wir haben GitHub. Und sonst entwickeln sie auf IA VIM Ja auf GitHub schauen wir ob ein Framework out of date ist oder nicht.

I: Wie geht ihr mit externen Open Source Anfragen um?

B: Inwiefern, ob sie das brauchen dürfen oder nicht?

I: Genau. Habt ihr dort einen Prozess? Wenn jetzt eine Anfrage kommt, wie beantwortet ihr diese?

B: Ja haben wir. Wir haben ein Portal, welche ich vorher gezeigt habe. Es kommen häufig anfragen, ob man unsere Codes benutzen darf und wieviel er kostet. Und wir müssen dann hinweisen, dass es ein offener ist und sie ihn benutzen können. Wir übernehmen keine warranty, ist explizit ausgeschlossen und steht im Lizenzfile drin. Wir fordern sie auf den Code zu benutzen. Wir gehen proaktiv vor. Wir binden sie auch ein, wie beim Crowdfunding Projekt. Schreiben wir sie an und sagen "hei ihr habt es ja von uns kopiert, wollt ihr jetzt nicht auch beitragen"? Doch es ist proaktiv. Ja wir fordern sie auf es zu benutzen.

I: Habt ihr Richtlinien wie ihr externe Open Source Komponente in eure Projekte integrieret?

B: Ja. Also wir haben dann noch den militärischen Teil und da gibt es vom ISBO eine Richtlinie, also das ist Informatiksicherheitsbeauftragter Bund. Vor allem wenn es in kritische Infrastrukturen reinkommt, dann scannen wir es durch und schauen was im Code drin ist. Gibt auch ein Review Prozess.

I: Und das scannen sie alles auf GitHub durch?

B: Teilweise wird das auch externalisiert, durch entsprechende Firmen, welche spezialisiert sind. Zum Beispiel haben wir jetzt QGIS. Für das Militär haben wir QGIS clients, ein Tool, um Karten anzuschauen, welches offline funktioniert im abgeschotteten Netz. Und das Ganze wird durch eine Sicherheitsabteilung gereviewed. Und das ist ja gerade auch der Vorteil von Open Source, weil da kann man es ja machen, weil der Source Code offen ist.

I: Unter welcher Lizenz veröffentlicht ihre Projekte generell?

B: Unter einer offenen. Du bist gut (seufzt) Weiss ich die Lizenz... Ich glaube es ist eine MIT Lizenz, muss es schnell nachschauen. Wir haben Apache Lizenz, BSD (...). Es gibt verschiedene Lizenzen. Non extensiv list. Die findest du da drin, kann ich dir noch geben.

I: Auf was achtet ihr bei der Lizenzauswahl?

B: Das sie möglichst offen und Änderungen wieder zurückgeben muss. Ja das es eine offene Lizenz ist.

I: Also permissive Lizenzen setzt ihr generell nicht ein?

B: Wo es möglich ist nicht, aber teilweise kann man es kaum umgehen. Also wir haben auch schon Komponente drin, bei 3D speziell. "Beggar can't be chooser". Ja es gibt nur diese Software und da kann man auch nicht darüber diskutieren.

I: Welche Software?

B: Zum Beispiel bei 3D. Also permissiv ist sie jetzt nicht. Sie hat eine Apache Lizenz und keine BSD Lizenz drin. Es ist immer noch eine offene Lizenz, aber sie ist permissiver.

I: Wie stellen sie sicher, dass Open Source Lizenzen eingehalten werden?

B: Also, wenn jemand versucht unser Produkt in unserem Namen zu verkaufen, dann schreiben wir den an und meistens ist es mit einem Email und einem Telefon geregelt. Wenn uns gemeldet wird, dass jemand unser Produkt verkaufen will.

I: Ist das schon vorgekommen?

B: Einmal gab es einen Versuch, aber das war vor Jahren. Das wurde schnell gelöst.

I: Und haben sie eine Rechtsabteilung mal involviert?

B: Ja die Rechtsabteilung ist involviert. Sie haben die Lizenzen gereviewed und uns einen Disclaimer hinzugefügt.

I: Welche Rolle spielt die Rechtsabteilung im Entwicklungsprozess?

B: Wenn wir Open Source Software extern entwickeln lassen, dann gibt es einen Werkvertrag und ein Forschungsvertrag, welcher die Rechtsabteilung noch anschaut. Und dort kommen die AGBs vom Bund bezüglich Open Source Software rein.

I: Wenn es auf GitHub ein Release gibt, wie entscheidet ihr, welche Projekte veröffentlicht werden?

B: Wir haben ein Change Board. Am Mittwoch hatten wir einen. Wir haben alle zwei Wochen einen neuen Release. Dort sagen die Entwickler gemeinsam was ein Release Kandidat und Publikationsbereit ist. Die internen Entwickler zusammen mit dem Projektleiter entscheiden dann.

I: Wann ist der beste Moment für eine Veröffentlichung?

B: Von Open Source Software?

I: Ja

B: Ist eine sehr gute Frage. Ich denke man muss einen Pain, also Schmerz haben. Das ist der grosse Vorteil von Open Source Software, wenn man agil vorgeht, DevOps. Dann kann man innerhalb von zwei bis drei Monate, nahe beim Schmerz, iterative Lösungen entwickeln. Also kann man schon früh shippen, also die ganze agile cycles nach dem Sprint. Ich denke, dass das wirklich das Gute ist, wenn man nach jedem Sprint, welcher man hat, wenn man agil vorgeht, Open Source Software veröffentlicht. Sehr schnell und früh. Auch ein nicht maturer Code, also ein Code, welcher nicht immer und überall funktioniert. Also so Alpha, Beta Code auch schon publiziert. Das ist dort die grösste Herausforderung für den Bund. Der Mindset muss sich ändern. Denn der Bund gibt eigentlich nur immer etwas Endgültiges raus. Zum Beispiel ist das Gesetz oder die endgültige finale Landeskarte. Und das ist ein Paradigmenwechsel, dass man sagt: "Komm wir veröffentlichen jetzt mal eine Alpha Version. Wir haben es auf Edge, Safari und Iphone getestet, aber auf Android noch nicht. Und schauen mal was die Leute sagen". Release fast, release early. Ist auch wichtig, wenn man im Bundesumfeld erfolgreiche Open Source Projekte haben will. Wenn man zu lange wartet, dann sieht meistens das UI veraltet aus und ist nicht mehr zeitgerecht. Dann ist auch die Akzeptanz bei Kunden nicht mehr so hoch.

I: Welche Parts vom Code werden veröffentlicht?

B: Der volle Source Code. Alle Source Code. Wir haben Teile vom Code, also wir haben Repos, welche wir nicht veröffentlichen. Und dort drin sind Sachen, wo absolut Swisstopo spezifisch und von internen Projekten sind. Aber alles was Public-facing von einem Auftrag vom Geo IG ist, veröffentlichen wir.

I: Sprechen wir über Partner. Arbeitet ihr mit Partner zusammen?

B: Ja. Wir arbeiten mit externen Auftragnehmern zusammen, welche unsere Codes teilweise weiterentwickeln. Dessen Mitarbeiter vielleicht Committer oder im project steering committee von einem Framework sind. Oder auch von einem Ein-Mann-Geschäft, welcher gewisse Komponente schreiben soll. Aber auch Industrie, welche sich im Open Source Bereich beteiligen und dort etwas weiterentwickeln. Ja wir arbeiten mit Partner zusammen. Auch mit anderen Kantonen, Fachhochschulen und Universitäten.

I: Wie wählt ihr eure Partner aus?

B: Wie bei normaler Beschaffung über das Einladungsverfahren oder Ausschreibungen. Und über das Netzwerk oder Konferenzen. Es ist halt wie die Community funktionieren

I: Ab welchem Zeitpunkt ist die Zusammenarbeit mit einem Partner sinnvoll?

B: (...) Die informelle Zusammenarbeit, um zu spüren was die Community will oder wo die Vision der Community hingeht, die kann schon sehr früh anfangen. Twitter kann, klingt sehr blöd / Twitter ist etwas sehr Gutes, um herauszufinden wo die Karawane hinzieht. Arbeitet man eher mit diesem Framework oder eher mit einem anderen Framework. Oder in Konferenzen. Man bringt sich schon sehr früh auch nur als Zuhörer in die Community ein. Es können auch Fachfremde sein. Um zu schauen wo es hingeht, bezüglich Lizenz, Frameworks, Funding oder Communityarten, welche man hat. Wenn man dann entwickeln will in Open Source, dann kommt das eigentlich bei uns relativ spät zu konkreten Projekten. Dann schreiben wir aus, gibt es ein Einladungsverfahren und werden die Komponente über Werk- oder Dienstleistungsverträge beschafft. Aber man ist frühe dabei. Aktiv eher ein wenig später.

I: Was ist der Grund wieso man Partner miteinbezieht und wieso entwickelt man es nicht selbst?

B: Ressourcen. Wir versuchen aber nur die Entwicklung von einer Komponente zu externalisieren. Das Ziel ist, dass wir immer selbst betreiben und Bugfixing selber machen können. Das ist wichtig wegen der Nachhaltigkeit. Das heisst, dass wir wirklich eine ganze Version mit internem Knowhow betreiben können, damit wir nicht abhängig von einem Dritten sind. Das macht es eben auch nicht billiger. Wenn man commercial of the shelf einkauft, dann kauft man sich das Wohlfühlpaket ein und dann hat man einen Supportvertrag, welcher kostet. Bei uns zahlt man halt intern Personenressourcen, welche Bugfixing machen. Da hat man natürlich eine personelle Volatilität. Da trägt man halt das Risiko mit. Wir investieren mehr in Leute als in Software.

I: Kommen wir zum Thema Schulung. Wie werden die Mitarbeiter im Umgang geschult und sensibilisiert?

B: Wir schauen das sie immer in Schulungen gehen, wir bieten das aktiv an. Sie werden auch weitergebildet. Diese Woche haben wir insgesamt zwei Mann Wochen geschult. Wir haben sehr viele Leute geschult in Provisionierung Framework, welches wirklich sehr tief im System unten ist. Wir schulen sie, aber wir beziehen sie auch aktiv mit ein, wenn wir nächsten Generationen oder Versionen von Lösungen planen. Fragen wir sie auch: " what is the newest shit". Sollen wir zum Beispiel auf D3js setzen bei Grafikdarstellungen und dann können sie mitentscheiden, ob das Sinn macht. Also Schulungen, bei strategischen Entscheiden mitbinden und dadurch bilden sich viele auch selber privat weiter. Muss man auch sagen. Es ist auch intrinsisch motiviert.

I: Sind das offizielle Trainings, welche sie anbieten?

B: Wir suchen diese Schulungen extern, bei zertifizierten Unternehmen, welche die Leute weiterbilden.

I: Und wer muss alles die Schulung machen?

B: Die Schulung können sie im Rahmen von der Weiterbildung machen. Manchmal gibt es obligatorische Schulung zum Beispiel, wenn wir ein neues Provisionierungssystem haben. Einführung in GitHub hat es auch schon gegeben, drei Tage Schulung GitHub. Ja die muss man dann machen.

I: Wenn man die von ihnen genannten Volatilität anschaut. Wie werden neue Mitarbeiter eingeführt? Gibt es da einen offiziellen Schulungsprozess?

B: Intern gibt es eine Einführung von Gesundheit bei swisstopo bis Nutzung email. Und da gibt es natürlich auch um das Thema wie man mit IT Mittel umgeht. Danach gibt es die Fachspezifischen Schulungen in den Abteilungen. Wenn man Open Source einsetzt, dann werden sie von Kollegen eingeführt, was unsere Best practice bei GitHub ist, mit dem Framework etc. Ja sind interne Vorgaben, welche wir haben.

I: Wie kann man das Lizenzrecht den Entwickler spannend erklären?

B: Keine Ahnung. Das ist den Entwickler meistens egal. Die kümmern sich häufig nicht darum. CTRL C und CTRL V. Ich muss ihnen sagen, dass man CTRL C, CTRL V nicht immer machen kann. Das ist ein grosses Problem. Am besten erklären kann es mit einem Email oder einem blick in die Inbox, weil jemand ein mail schreibt und unglücklich ist, weil man sein Piece of code gebraucht hat. Dazu habe ich keine Lösung. Wir lernen immer nur, wenn jemand etwas kopiert, wo eigentlich eine Lizenz drauf wäre.

I: Musstet ihr auch schon zwangsveröffentlichen?

B: Nein

I: Mit was für Communities arbeitet ihr zusammen?

B: Thematisch mit der Geo Community, das ist fachlich die OGC, also Open Geospatial Consortium (Beoachtend), mit welcher wir zusammenarbeiten. Technisch auf Infrastukturlevel mit Netzwerk Debian (Beoachtend), also mit Betriebssystem Community. PostGres QGIS (Beoachtend)) Community, Openlayers (activ), Angular auf der Serviceebene (Beoachtend). Und im Frontend mit der Javascript Community mit welcher wir zusammenarbeiten.

I: Wer ist Teil von diesen Communities? Also sind es private Entwickler oder Unternehmen?

B: Unternehmer und private Entwickler. Die grossen ist zb Boundless, das sind die aus dem US militärisch industriellen Komplex. Firmen und Spinoffs welche dort arbeiten. Es ist sehr international, aber auch in der Schweiz gibt es eine Community.

I: Haben sie eine selber aufgebaut oder seid ihr bei mehreren beigetreten?

B: Wir haben selbst eine aufgebaut. Das ist API3 Google Groups. Also api3.geo.admin.ch findest du auf der Startseite. Wir haben diese im Sinne User helfen Nutzer. Das ist die wir selbst aufgebaut haben und da sind ungefähr 400 Leute drin. Wenn sie ein Problem in unserem Framework sehen, dann schreiben sie dort rein. SoMe: Twitter Community haben wir zusammen mit dem GeoWebforum, das ist nicht nur Software, sondern mehr. Twitter ist ja nicht nur Software, aber häufig Software. Sonst klinken wir uns ein, Openlaysers Community ein vor allem oder via GitHub.

I: Wie seid ihr beim Aufbau eurer Community vorgegangen?

B: Trial-and-Error. Einfach Leute angeschrieben, telefoniert, Mailinglisten aus Projekten erstellt, Schulen, Fachhochschulen oder Konferenzen in der Schweiz. Eben die Schweiz ist klein. Wir haben eine sehr föderale Struktur und viele Kantone haben gleiche Probleme. Viele Gemeinde auch. Dann setzt man sich zusammen an einen Tisch. Plötzlich sieht man, dass man das gleiche Framework benötigt. Sendet dann ein Mail, fügte noch einen mehr auf die Empfänger Liste drauf, so entsteht eine Mailinglist und es wurden immer mehr Leute involviert. An der nächsten Konferenz war es dann einen ganzen Block zu diesem Thema. Ist organisch gewachsen und war rein interessengetrieben, intrinsisch.

I: Was ist euer Beitrag an die Community?

B: Finanziell, sicher auch. Das wir sagen: "Hei wir machen eine Anstossfinanzierung. Wir geben X Franken, wenn ihr auch noch weitere Y Franken findet, das macht zusammen Z Franken". Wir koordinieren noch, ist in unserem Auftrag, der Koordinationsauftrag. Wir helfen auch Standards zu entwickeln und versuchen Risikoabschätzungen zu machen, Abklärungen zu machen, ob man das einsetzten darf oder nicht. Dann noch Kompetenzen aufbauen, Schulungen zu bilden und koordinieren. Das ist so ein bisschen unser Beitrag.

I: Und was habt ihr bei den Beitritten zu Communities beachtet? Wie habt ihr diese Communities gefunden?

B: Sie müssen genügend gross, nachhaltig, also eine kritische Grösse aufweisen. Sie muss eine offene Community sein und auch eine Zukunft haben. Das ist so das wichtigste. International, keine geschlossene Community (...).

I: Welche Art von Beiträgen erwartet ihr dann von einer Community?

B: Es ist einerseits, dass sie sich finanziell beteiligen, wenn sie Code Sprints geben. Sie strategische oder operative Entscheidungen treffen. Weiterentwicklungen machen. Das sie offen sind, also "no one left behind". Ist das klassische Manifest, welches man hat. Und da muss man schon aufpassen. Gibt verschiedene Communities und Frameworks. Also Offenheit, Bereitschaft zu investieren, also auch mit Manpower. Dass die Community intrinsisch motiviert ist. Ja das ist so das wichtigste.

I: Betreiben sie ein aktives Community Management?

B: Ja kann man so sagen. In verschiedenen Foren. Wir haben verschiedene Foren auch aufgebaut, eben API3 Google Groups Foren. Wir melden Bugs über GitHub zurück. Öffentlich Bugs kommunizieren und die Leute ansprechen. Wir ermutigen auch private dazu, bei uns Bugfixing zu machen, sie dürfen gerne contributen.

I: Haben sie auch schon externe Entwickler bezahlt?

B: Ja über Werks- oder Forschungsverträge.

I: Haben sie bei der Community, welche sie aufgebaut haben, eine Community Governance

B: Nein. Bei der Softwareentwicklung, weil wir Software nicht immer allein entwickeln, dort haben wir eine Governance, im Sinn, wenn wir einen pull request akzeptieren. Dann reviewen wir den pull request und ein interner Senior muss ihn freigeben.

I: Wie viele interne Senioren arbeiten bei ihnen?

B: Zwei.

I: Wir gehen sie mit der Balance zwischen der Kontrolle und Offenheit mit der Community um?

B: Ja, dass man Fehlertolerant ist, wenn man Fehler findet, dann fixed man sie. Es gibt eine rote Linie, die man nicht überschreitet, wie Username, Passwort oder etc. hardcodiert in Code oder solche Sachen. 2FA da sind wir ziemlich stringent, wenn so etwas vorkommt.

I: Wie ist die Entscheidungsfindung? Zentralisiert oder dezentralisiert?

B: Trotz allem eher zentralisiert, ist eigentlich eigentlich gegen den Community gedanke.

I: Und wie nimmt dies die Community auf?

B: Es ist schon so, dass es häufig so ist, dass wer bezahlt, der befiehlt. Und dann gibt es eben auch diese verschiedenen Diven in diesen Communities. Dann gab es auch schon ganze Bereiche in diesen Communities gegeben, welche dann gesagt haben, dass sie halt aus einer Community austreten. Es "menschelet", ist leider schon so.

I: Ich habe auf eure Website die Dienstleistungen gesehen. Bei nicht publizierten geologischen Informationen.

B: Das sind Daten.

I: Welche Daten werden jetzt öffentlich? Hier haben wir ja public Service Daten?

B: Von Swisstopo sind es ungefähr 20 bis 30 Datensätze, viele geologische Datensätze. Allgemein interessant für Strassennamen, Adressverzeichnis, Grenzen und Ortsflurnamen. Hintergrunddaten wie ZB Landeskarte ist noch eine Diskussion im Gang, wann das kommt. Da muss man uU das Gesetz noch ändern. Auf Opendata Swiss findet man fast 300 bis 400 GeoDatensätze welche offen sind, welche wirklich frei sind. Also mit frei meine ich, dass sie einer CC Lizenz, nicht nur Open Access sind.

I: Bei Geodaten und Geodienste habe ich Lizenzen gefunden.

B: Swisstopo hat auch noch Software, welche sie selbst entwickeln. Die sind geschlossen und das ist Closed Source. Wir haben nicht nur Open Source Software. Beim Geoportal ist alles offen. Es gibt geodätische Software, welche Closed ist und es gibt kostenpflichtige Software.

I: Das ist eher bei den geodätischen Softwares der Fall?

B: siehe https://www.swisstopo.admin.ch/de/home/meta/konditionen/geodaetische-software.html.

I: Alles was wir auf dieser Seite sehen, ist nicht auf GitHub zu finden?

B: Nein das ist Swisstopo und auf GitHub ist nur von geo.admin.ch. Und bei Swisstopo haben einzelne kleine Projekte einen Github Account. Swisstopo hat auch noch geschlossene Software. Es ist nicht alles was wir machen offen.

I: Meistens ist hier die Nutzung trotzdem offen. Aber die Veröffentlichung für kommerzielle Zwecken nicht?

B: Ja von Fall zu Fall unterschiedlich.

I: Dann habe ich auch noch gelesen, dass die Megapixel, welche man nutzt, in Rechnung gestellt werden.

B: Geht hier nicht um OSS, sondern um Open Data- Ja das ist auf Freemiumaproach. Anschauen kann man das meiste gratis, aber um es als Geodienst zu nutzen, hat es eine Paywall drin.

I: Das war für mich ein wenig verwirrend, weil ich es zuerst auf GitHub gesehen habe.

B: Da bist du nicht der Erste. Das ist bei allen so. Man hat ein BFE und ein BAV, Bundesamt für Energie und Verkehr, welche eine komplette offenere Datenstrategie haben. Man hat Swisstopo, welche die Daten über das Geoportal Bund, welches offen ist, vertreiben. Swisstopo hat aus rechtlichen Gründen nicht alles offen, weil die Verordnung es vorschreibt, dass wir etwas verlangen müssen. Die Swisstopo Verordnung ist in Bearbeitung, dass es auch mal uU frei wird.

I: Habe noch gesehen, dass die Schulen möglichst alles nutzten können?

B: Ja.

I: Wenn ich Swisstopo GitHub eingebe, dann kommt man auf geo.admin.

B: Wirklich, das habe ich noch nicht geprüft. Wir vertreiben es. Das ist eben genau der Punkt. Swiss Geoportal ist eben mehr als nur die Swisstopo.

I: Geoportal ist Open Source und dann gibt es noch die Swisstopo.

B: Das ist eine andere Seite. Du muss auf www.geo.admin.ch gehen oder GitHub. Schicke dir noch den Link.

I: GitHub habe ich angeschaut. Dann habe ich gleich noch Swisstopo eingegeben und mich gefragt, ob es nun wirklich Open Source ist.

B: (Lacht)

I: Das was hier unter Lizenzbestimmungen besteht ist was anderes.

B: Das ist ganz etwas anders. Lizenzbedingungen von Daten und nicht von SW

I: Und seid ihr dort auch involviert, bei Geodätische Daten?

B: Nein das sind die Entwickler. Die machen zum Teil auch Spezialsoftware für Dritte, welche man dann abrechnen muss.

I: Ich habe noch eine Frage. Es gibt da ein Maturitätsmodel von der Eclipse Fundation. Wo würdet ihr euch hier einschätzen?

B: Wir sind beim zwei.

I: Beim Contribute?

B: Wir sind beim Contribute. Die Crowdfunding Kampagne welches wir machen ist schon eher beim vier. Ja man könnte auch drei sagen. Wir haben unsere Software, wenn du das Eclipse Model anschaust, ist die Wiederverwendung oder. Nein wir sind beim drei. Unsere Software wird an sehr vielen Orten wiederverwendet und wir investieren auch teilweise. Dann gibt es ab und zu Phasen, wo wir sagen, dass es wichtig ist das wir Open Source machen, damit andere es auch nutzen können und jetzt investieren wir strategisch. Das ist zwischen drei und vier. Schon lange nicht mehr gesehen.

I: Danke vielmal für eure Zeit.

B: Ja bin sehr gespannt auf die Studie.

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

BSD	Berkeley Software Distribution (License)
FSFE	Free Software Foundation Europe
GPL	GNU General Public License
OSS	Open Source Software
OS-Lizenzen	Open Source Lizenzen

Literaturverzeichnis

Ågerfalk, & Fitzgerald. (2008). Outsourcing to an Unknown Workforce: Exploring Opensurcing as a Global Sourcing Strategy. *MIS Quarterly*, *32*(2), 385–409.

Andersen-Gott, M., Ghinea, G., & Bygstad, B. (2012). Why do commercial companies contribute to open source software? *International Journal of Information Management*, *32*(2), 106–117.

Baloise. (2019). Open Source @ Baloise. Abgerufen 4. Juni 2019, von https://github.com/baloise/open-source

Baloise.ch. (2019). Porträt. Abgerufen 5. Juni 2019, von https://www.baloise.ch/de/ueber-uns/portraet.html

Baloise.com. (2019a). Auf einen Blick. Abgerufen 5. Juni 2019, von https://www.baloise.com/de/home/ueber-uns/wer-wir-sind/auf-einen-blick.html

Baloise.com. (2019b). Strategie. Abgerufen 5. Juni 2019, von https://www.baloise.com/de/home/ueber-uns/wofuer-wir-stehen/strategie.html

Bonaccorsi, A., & Rossi, C. (2006). Comparing motivations of individual programmers and firms to take part in the open source movement: From community to business. *Knowledge, Technology & Policy*, *18*(4), 40–64.

Carbone, P. (2007). Competitive Open Source. *Open Source Business Resource*, (July 2007). Abgerufen von https://timreview.ca/article/93

Dahlander, L. (2005). Appropriation and appropriability in open source software. *International Journal of Innovation Management*, *09*(03), 259–285.

Dahlander, L., & Magnusson, M. (2008). How do Firms Make Use of Open Source Communities? *Long Range Planning*, *41*(6), 629–649.

Dahlander, L., & Magnusson, M. G. (2005). Relationships between open source software companies and communities: Observations from Nordic firms. *Research Policy*, *34*(4), 481–493.

Dernbach, C. (2002). Microsoft benennt Linux als Konkurrent Nr. 1. Abgerufen 19. März 2019, von https://www.heise.de/newsticker/meldung/Microsoft-benennt-Linux-als-Konkurrent-Nr-1-68685.html

Ebert, C. (2007). Open Source Drives Innovation. *IEEE Software*, *24*(3), 105–109.

Eclipse Foundation. (2009). *The Open Source Developer Report 2009 Eclipse Community Survey*. Abgerufen von https://www.eclipse.org/org/press-release/Eclipse_Survey_2009_final.pdf

Eisenhardt, K. M. (1989). Building Theories from Case Study Research. *The Academy of Management Review*, *14*(4), 532–550.

Finanzdirektion. (2018). Weniger abhängig von Software-Anbietern: Der Kanton schafft Transparenz mit Open Source-Software. Abgerufen 5. Juni 2019, von https://www.be.ch/portal/de/index/mediencenter/medienmitteilungen/suche.archiv.meldungNeu.html/portal/de/meldungen/mm/2018/12/20181211_1603_der_kanton_schaffttransparenzmitopensource-software.html

fin.be.ch. (2019). Broschüre „Herzlich willkommen im KAIO“. Abgerufen 5. Juni 2019, von https://www.fin.be.ch/fin/de/index/direktion/organisation/kaio.assetref/dam/documents/FIN/KAIO/de/1_Die-Direktion_Organisation/Brosch%C3%BCre_KAIO_de.pdf

Flyvbjerg, B. (2006). Five Misunderstandings About Case-Study Research. *Qualitative Inquiry*, *12*(2), 219–245.

Free Software Foundation Europe. (2019). *Public Money Public Code – Modernising Public Infrastructure with Free Software*. 32. Abgerufen von https://download.fsfe.org/campaigns/pmpc/PMPC-Modernising-with-Free-Software.pdf

FSFE. (2019). Public Money, Public Code. Abgerufen 12. Mai 2019, von https://publiccode.eu/

GitHub and Government. (2019). Who's using GitHub? |. Abgerufen 26. Mai 2019, von https://government.github.com/community/

GitHub Kanton Bern. (2019). Kanton Bern. Abgerufen 5. Juni 2019, von https://github.com/kanton-bern

Glätti, B. (2014). Wie kann die Freigabe von Open-Source-Software durch die Bundesverwaltung explizit erlaubt werden? | Geschäft | Das Schweizer Parlament. Abgerufen 5. Juni 2019, von https://www.parlament.ch/de/ratsbetrieb/suche-curia-vista/geschaeft?AffairId=20144275

Glynn, E., Fitzgerald, B., & Exton, C. (2005). Commercial adoption of open source software: an empirical study. *International Symposium on Empirical Software Engineering*, 217–226. Queensland, Australia: IEEE.

gnu.org. (2019). Freie Software. Was ist das? Abgerufen 19. März 2019, von https://www.gnu.org/philosophy/free-sw

Goldstein, A. (2018). Top 10 Open Source Licenses in 2018: Trends and Predictions. Abgerufen 25. Mai 2019, von https://resources.whitesourcesoftware.com/blog-whitesource/top-open-source-licenses-trends-and-predictions

Grand, S., von Krogh, G., Leonard, D., & Swap, W. (2004). Resource allocation beyond firm boundaries. *Long Range Planning*, *37*(6), 591–610.

Greene, T. C. (2001). Ballmer: „Linux is a cancer“. Abgerufen 19. März 2019, von https://www.theregister.co.uk/2001/06/02/ballmer_linux_is_a_cancer/

Greve, G. C. F. (2016). *Open-Source-Software 2.0 Leitfaden*. Berlin: Bitkom e.V.

Gulley, N., & Lakhani, K. R. (2010). *The Determinants of Individual Performance and Collective Value in Private-Collective Software Innovation* (Working Paper Nr. 10–065). Boston: Harvard Business School Technology & Operations Mgt.

Hauge, Ø., Ayala, C., & Conradi, R. (2010). Adoption of open source software in software-intensive organizations – A systematic literature review. *Information and Software Technology*, *52*(11), 1133–1154.

Helvetia.com. (2019). Wer wir sind | Helvetia Gruppe. Abgerufen 5. Juni 2019, von https://www.helvetia.com/corporate/web/de/home/ueber-uns/uebersicht/wer-wir-sind.html

Hippel, E. von, & Krogh, G. von. (2003). Open Source Software and the “Private-Collective” Innovation Model: Issues for Organization Science. *Organization Science*, *14*(2), 209–223.

Informatiksteuerungsorgan des Bundes. (2019). IKT-Teilstrategie OpenSource. Abgerufen 5. Juni 2019, von https://www.isb.admin.ch/isb/de/home/ikt-vorgaben/strategien-teilstrategien/sb004-ikt-teilstrategie-open-source.html

inside-it.ch. (2011). Open Justitia wird tatsächlich open. Abgerufen 26. Mai 2019, von https://www.inside-it.ch/articles/26217

Jost, Battagliero, Kohli, Sancar, & Sollberger. (2013). *Motion 177-2013 Synergien beim Software-Einsatz im Kanton Bern nutzen* (Motion Nr. 177–2013; S. 6). Bern: FIN.

Lerner, J., & Tirole, J. (2003). Some Simple Economics of Open Source. *The Journal of Industrial Economics*, *50*(2), 197–234.

Lerner, J., & Tirole, J. (2005). The Economics of Technology Sharing: Open Source and Beyond. *Journal of Economic Perspectives*, *19*(2), 99–120.

Mayring, P. (2014). *Qualitative content analysis: theoretical foundation, basic procedures and software solution*. Klagenfurt: URN.

Microsoft News Center. (2018, Juni 4). Microsoft to acquire GitHub for $7.5 billion. Abgerufen 19. März 2019, von https://news.microsoft.com/2018/06/04/microsoft-to-acquire-github-for-7-5-billion/

Müller, G., & Vogel, S. (2014). *Rechtsgutachten zur verfassungsrechtlichen Zulässigkeit der Randnutzung von Software im Verwaltungsvermögen, insbesondere der Veröffentlichung und Verbreitung von Open-SourceSoftware durch Träger von Bundesaufgaben*. Abgerufen von https://www.news.admin.ch/NSBSubscriber/message/attachments/37015.pdf

O'Mahony, S. (2007). The governance of open source initiatives: what does it mean to be community managed? *Journal of Management & Governance*, *11*(2), 139–150.

Open Justitia. (2019). Die Community umfasst die folgenden Mitglieder. Abgerufen 4. Juni 2019, von http://www.openjustitia.ch/DE/00_Mitgliedschaften_OpenJustitia_d.pdf

Opensource.org. (2019a). Open Source Initiative History of the OSI. Abgerufen 19. März 2019, von https://opensource.org/history

Opensource.org. (2019b). Open Source Initiative Licenses by Name. Abgerufen 26. Mai 2019, von https://opensource.org/licenses/alphabetical

Opensource.org. (2019c). Open Source Initiative The Open Source Definition. Abgerufen 26. Mai 2019, von https://opensource.org/osd

Osterloh, M., Rota, S., & Kuster, B. (2004). Open-Source-Softwareproduktion: Ein neues Innovationsmodell? *Open Source Jahrbuch 2004*, 121–138.

Perler, L. (2018, Juni 4). Microsoft kauft GitHub für 7,5 Milliarden Dollar. Abgerufen 19. März 2019, von https://www.computerworld.ch/business/uebernahme/microsoft-kauft-github-7-5-milliarden-dollar-1542840.html

Pizka, M., & Bauer, A. (2004). A brief top-down and bottom-up philosophy on software evolution. *Proceedings. 7th International Workshop on Principles of Software Evolution, 2004.*, 131–136. Kyoto, Japan: IEEE.

Poledna, T., Schlauri, S., & Schweizer, S. (2017). *Rechtliche Voraussetzungen der Nutzung von Open-SourceSoftware in der öffentlichen Verwaltung, insbesondere des Kantons Bern*. Berlin – Bern: Carl Grossmann.

Porter, M. E. (1996). What is Strategy. *Harvard Business Review*, *74*(6), 61–78.

Raymond, E. (1999). The cathedral and the bazaar. *Knowledge, Technology & Policy*, *12*(3), 23–49.

Raymond, E. (o. J.). *The cathedral and the bazaar*. 27.

Regierungsrat des Kantons Bern. (2018). Verordnung über die Informations- und Telekommunikationstechnik der Kantonsverwaltung (ICTV). Abgerufen 5. Juni 2019, von https://www.belex.sites.be.ch/frontend/versions/1410

Schrape, J.-F. (2016a). *Open-Source-Projekte als Utopie, Methode und Innovationsstrategie*. Glückstadt: Hülsbusch.

Schrape, J.-F. (2016b). *Open-Source-Projekte als Utopie, Methode und Innovationsstrategie: historische Entwicklung - sozioökonomische Kontexte -*

Typologie. Glückstadt: vwh, Verlag Werner Hülsbusch, Fachverlag für Medientechnik und -wirtschaft.

Schrape, J.-F. (2017). Open-Source-Projekte als utopischer Gegenentwurf, Entwicklungsmethode und Innovationsstrategie. *Kongress der Deutschen Gesellschaft für Soziologie*, *38*, 1–12.

Schrape, J.-F. (o. J.). *Open-Source-Projekte als utopischer Gegenentwurf, Entwicklungsmethode und Innovationsstrategie*. 12.

Schweizerische Eidgenossenschaft. (2017). *Freigabe von Open-Source-Software durch die Bundesverwaltung - Bericht des Bundesrates in Erfüllung des Postulats Balthasar Glättli vom 12. Dezember 2014 (14.4275)* (S. 10) [Bericht des Bundesrates in Erfüllung des Postulats Balthasar Glättli vom 12. Dezember 2014 (14.4275)]. Abgerufen von https://www.parlament.ch/centers/eparl/curia/2014/20144275/Bericht%20BR%20D.pdf

Sen, R., Subramaniam, C., & Nelson, M. L. (2008). Determinants of the Choice of Open Source Software License. *Journal of Management Information Systems*, *25*(3), 207–240.

Six-group.com. (2019). SIX im Überblick. Abgerufen 5. Juni 2019, von https://www.six-group.com/de/home/company/overview-of-six.html

SRF Data. (2019a). Code & data from SRF Data, the data-driven journalism unit of Swiss Radio and TV. Abgerufen 4. Juni 2019, von https://github.com/srfdata

SRF Data. (2019b). SRF Data auf GitHub. Abgerufen 5. Juni 2019, von https://srfdata.github.io/

Srf.ch. (2019). Unternehmensporträt - Dafür stehen wir. Abgerufen 5. Juni 2019, von https://www.srf.ch/unternehmen/unternehmen/portraet/unternehmensportraet-dafuer-stehen-wir

Stallman, R. (2009). Viewpoint Why open source misses the point of free software. *Communications of the ACM*, *52*(6), 31–33.

Stuermer, M. (2009). *How Firms Make Friends: Communities in Private-Collective Innovation* (Doctoral Dissertation). ETH Zürich, Zürich.

Stürmer, M., & Gauch, C. (2018). *Open Source Studie 2018*. Bern: swissICT, CH Open.

Swiss Geoportal. (2019). github.com/geoadmin. Abgerufen 4. Juni 2019, von https://github.com/geoadmin

swisstopo.admin.ch. (2019a). Organisation. Abgerufen 5. Juni 2019, von https://www.swisstopo.admin.ch/de/swisstopo/organisation.html

swisstopo.admin.ch. (2019b). Vision und strategische Stossrichtungen. Abgerufen 5. Juni 2019, von https://www.swisstopo.admin.ch/de/swisstopo/vision.html

Urech, M. (2015). Bundesrat prüft Open-Source-Freigaben durch Bundesstellen. Abgerufen 5. Juni 2019, von https://www.netzwoche.ch/news/2015-03-23/bundesrat-prueft-open-source-freigaben-durch-bundesstellen

Ven, K., & Verelst, J. (2006). The Organizational Adoption of Open Source Server Software by Belgian Organizations. In E. Damiani, B. Fitzgerald, W. Scacchi, M. Scotto, & G. Succi (Hrsg.), *Open Source Systems* (Bd. 203, S. 111–122). https://doi.org/10.1007/0-387-34226-5_11

von Krogh, Haefliger, Spaeth, & Wallin. (2012). Carrots and Rainbows: Motivation and Social Practice in Open Source Software Development. *MIS Quarterly*, *36*(2), 649–676.

Weibel, T. (2012). Freigabe von Open-Source-Software durch Behörden. Abgerufen 3. Juni 2019, von https://www.parlament.ch/de/ratsbetrieb/suche-curia-vista/geschaeft?AffairId=20124247

WhatIs.com. (2019). vendor lock in. Abgerufen 26. Mai 2019, von https://whatis.techtarget.com/search/query?q=vendor+lock+in+

Wichmann, T. (2002). *Firms' Open Source Activities - Motivations and Policy Implications* [Survey and Study]. Free/Libre and Open Source Software (FLOSS).

Yin, R. K. (2003). *Case study research: design and methods.* (2. Aufl.). Thousand Oaks: SAGE.

Zellweger, C., & Preisig, S. (2018, September 3). Kanton will seine starke Abhängigkeit verringern. *www.derbund.ch*. Abgerufen von https://www.derbund.ch/bern/kanton/kanton-bern-will-abhaengigkeit-bei-informatik-verringern/story/15462565

Selbständigkeitserklärung

„Ich erkläre hiermit, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen benutzt habe. Alle Stellen, die wörtlich oder sinngemäss aus Quellen entnommen wurden, habe ich als solche gekennzeichnet. Mir ist bekannt, dass andernfalls der Senat gemäss Artikel 36 Absatz 1 Buchstabe o des Gesetzes vom 5. September 1996 über die Universität zum Entzug des aufgrund dieser Arbeit verliehenen Titels berechtigt ist.“

Handschriftliche Unterschrift

Bern, Datum Vorname Name

Veröffentlichung der Arbeit

I.d.R. werden schriftliche Arbeiten öffentlich zugänglich gemacht.

- ☐ Hiermit erlaube ich, meine Arbeit auf der Website der Forschungsstelle Digitale Nachhaltigkeit zu veröffentlichen.
- ☐ Ich möchte auf eine Veröffentlichung meiner Arbeit verzichten.

Falls eine Vertraulichkeitserklärung unterschrieben wurde, ist es Sache des Studierenden, das Einverständnis des Praxispartners einzuholen. Es muss der Arbeit eine schriftliche Bestätigung des Praxispartners beigelegt werden.

Die Benotung der Arbeit erfolgt unabhängig davon, ob die Arbeit veröffentlicht werden darf oder nicht.

Handschriftliche Unterschrift

Bern, Datum Vorname Name