/ soziales

Digitale Solidarökonomie

Open-Source-Softwareentwicklung und ihre Potentiale für die Soziale Arbeit

Clemens Rosenthaler
Matr. Nr.: 1510406028

Bachelorarbeit 2
Eingereicht zur Erlangung des Grades
Bachelor of Arts in Social Sciences
an der Fachhochschule St. Pölten

Datum: 16.04.2018
Version: 1

Begutachter:
Thomas Truppe BA, MA
Mag. DSA Christian Tuma

Abstract

Open-Source-Software wird häufig von freiwilligen Entwickler*innen kooperativ und selbstorganisiert programmiert und den Nutzer*innen kostenlos zur Verfügung gestellt. Wie diese Form der digitalen Solidarökonomie in der Sozialen Arbeit genutzt werden könnte, war die zentrale Fragestellung dieser Bachelorarbeit. Diesbezüglich wurden Daten mittels Interviews mit drei Experten aus dem technischen wie auch sozialwissenschaftlichen Kontext erhoben und durch die qualitative Inhaltsanalyse nach Mayring ausgewertet. Zum einen zeigte sich Potential für die Nutzung von Open-Source-Software in der Kostenfreiheit, die sowohl für Klient*innen als auch für Organisationen der Sozialen Arbeit eine Ressource darstellt. Zum anderen wird aufgezeigt, wie durch dieses Konzept verschiedene Akteur*innen der Sozialen Arbeit in die Entwicklung fachspezifischer Software eingebunden werden können.

Open-source-software is commonly developed by voluntary programmers on the basis of cooperation and self-organisation and is provided to the users free of charge. The aim of this bachelor thesis is to answer the question of how this kind of solidary economy could be used in social work. To do so, data was collected via interviewing three experts from a technical as well as a social-scientific context. The data was evaluated according to Mayring`s qualitative content analysis. On the one hand, open-source-software provides a resource for clients as well as organisations of the social work due to the free costs. On the other hand, open source could be a concept to incorporate different stakeholders of the social work in the development of subject-specific software.

Inhalt

1. Einleitung

Soziale Arbeit, Solidarökonomie und Digitalisierung – drei Begriffe, zu denen es jeweils einen reichhaltigen Diskurs gibt. Demgegenüber ist eine Debatte, die diese drei Themenfelder zu verbinden versucht, kaum vorzufinden. Um hier Berührungspunkte aufzuzeigen, bedarf es einer näheren Untersuchung. Nicht, weil diese nicht vorhanden sind, sondern vielmehr deshalb, weil sich die Blicke auf diese drei unterschiedlichen Thematiken im sozialarbeiterischen Diskurs selten kreuzen. In der vorliegenden Bachelorarbeit soll eine solche Verbindung aufgezeigt werden. Im Zuge des Bachelorprojektes „Solidarökonomie" des Studienganges Soziale Arbeit an der Fachhochschule St. Pölten wurden verschiedene solidarökonomische Projekte beforscht. Das Interesse an diesen Initiativen war breit angelegt und umfasste neben der Entstehung, den Mitgliedern und den Konzepten vor allem die Nutzungsmöglichkeiten für die Soziale Arbeit.

Solidarökonomische Projekte kennt man vor allem in der „analogen Welt" in Form von Gemeinschaftsgärten, Reparatur-Cafés, Tauschkreisen etc. Im Großteil dieser Initiativen hat die Digitalisierung bereits Einzug gehalten, in dem sich die „analogen Projekte" in ihrer praktischen Tätigkeit digitaler Medien bedienen. Ein anschauliches Beispiel dafür sind Tauschkreise, die zur Verrechnung ihrer eigenen Währung digitale Plattformen, ähnlich dem Online-Banking, verwenden (vgl. Talenteverbund o.A.). Demgegenüber gibt es solidarökonomische Projekte, die es nur in der „digitalen Welt" des Internets gibt und gerade aus dieser Umwelt ihre Potentiale schöpfen. Durch die technische Infrastruktur des Internets können die Grenzen von Zeit und Raum der „analogen Welt" – zumindest teilweise – aufgehoben werden. Menschen können sich unabhängig von ihrem Aufenthaltsort weltweit vernetzen und Informationen über Jahre hinweg speichern und rund um die Uhr zugänglich machen. Eines der wohl bekanntesten Beispiele für ein digitales solidarökonomisches Projekt stellt Wikipedia dar. Die Online-Enzyklopädie hat ihre Fülle an Artikeln der weltweiten Beteiligung von freiwilligen Mitarbeiter*innen zu verdanken, die – vernetzt durch das Internet – in kollaborativen Prozessen Inhalte verfassen, diskutieren und kontrollieren (vgl. Wikipedia 2018).

Thema dieser Bachelorarbeit ist die *Open-Source-Softwareentwicklung*, die sich ebenso wie Wikipedia als eine Form des kollaborativen Schreibens verstehen lässt. Jedoch handelt es sich hierbei nicht um das Verfassen einer Enzyklopädie, sondern um das „Schreiben" von Software. Der entstehende Programmcode[1] wird, anders als beim Großteil der kommerziellen Software, nicht

1 Der Übersichtlichkeit halber wird erst an späterer Stelle, im Kapitel 2.1.3., die Bedeutung der Begriffe Software und Programmcode erläutert.

geheim gehalten, sondern veröffentlicht, damit dieser verändert, weiterentwickelt oder damit gelernt werden kann. Das Open-Source-Konzept sieht vor, dass Software nicht ausschließlich durch ein abgegrenztes Entwickler*innen-Team programmiert wird, sondern Interessent*innen und Nutzer*innen in den Entwicklungs- und Programmierungsprozess mit eingebunden werden und die Software frei zur Verfügung gestellt wird.

Um einen Zusammenhang zwischen Sozialer Arbeit, Solidarökonomie und Digitalisierung herzustellen, ging ich für meine Arbeit von den folgenden beiden Überlegungen aus: *Erstens* scheinen sich die Prinzipien und Werte der Solidarökonomie mit jenen der Sozialen Arbeit zu überschneiden. In solidarökonomischen Projekten wird bestrebt, dass die Mitglieder in einem gemeinsamen Prozess des Organisierens und Kooperierens Dienstleistungen oder Produkte hervorbringen und sich dadurch gegenseitig unterstützen. Dieser Gedanke der Partizipation sowie die Überlegung, einen Zugang zu Ressourcen zu schaffen, zeugen von der Anschlussfähigkeit an die Soziale Arbeit und werfen die Frage auf, ob und wie sich solche Projekte in diesem Kontext nutzen lassen. *Zweitens* ist Soziale Arbeit längst keine rein „analoge“ Arbeit mehr. Die Digitalisierung zeigt sich in verschiedensten Formen: Klient*innenakten werden elektronisch angelegt, Einrichtungen veröffentlichen ihre Angebote und Informationsmaterial im Internet, E-Mail hat sich als Kommunikationsmittel etabliert, in Beratungen gibt es die Möglichkeit des Videodolmetschens. Darüber hinaus werden die Lebenswelten und der Sozialraum der Menschen um eine digitale Dimension erweitert. Mit all den Möglichkeiten, die sich daraus ergeben, entstehen auch neue Fragestellungen für die Soziale Arbeit.

Der Fokus dieser Bachelorarbeit liegt auf den Nutzungsmöglichkeiten der Open-Source-Softwareentwicklung für die Soziale Arbeit. In der vorliegenden Arbeit werden somit die Themenfelder der Solidarökonomie und der Digitalisierung im sozialarbeiterischen Kontext verbunden. Die Arbeit ist im Wesentlichen in fünf Teile gegliedert: Nach der Einleitung folgen theoretische Grundlagen, in denen die für die Arbeit wesentliche Begriffe definiert werden, das Konzept von Open Source und dessen Verbindung zur Solidarökonomie dargelegt und die Digitalisierung im Rahmen der Sozialen Arbeit beleuchtet werden. Danach wird anhand dieser Grundlagen eine Forschungsfrage formuliert und das Design der empirischen Forschung erläutert. Nach der Darstellung der Ergebnisse werden diese in Hinblick auf die Forschungsfrage zusammengefasst und diskutiert. Die darauffolgende kritische Reflexion dieser Arbeit soll auf Implikationen hinweisen und einen Ausblick auf weitere Forschung geben.

2. Theoretische Grundlagen

Dieses Kapitel soll den Kontext der Forschung näher erläutern. Ziel ist es, relevante Begriffe zu erklären und die Verbindungen zwischen Solidarökonomie, Open Source und Sozialer Arbeit herauszuarbeiten.

2.1. Relevante Begriffe

2.1.1. Solidarökonomie

Der Begriff der Solidarökonomie wird im Diskurs unterschiedlich verstanden. Die fehlende einheitliche Begriffsdefinition könnte darauf zurückgeführt werden, dass solidarökonomische Projekte dezentral, als Bottom-Up-Prozesse, organisiert sind und es daher zu verschiedenen Auslegungen des Begriffes kommt (vgl. Attac o.A.). Unterschiedliche Definitionen haben gemein, dass Solidarökonomie einen Gegenentwurf zum kapitalistischen Wirtschaftssystem darstelle und anstatt von Gewinnmaximierung andere, soziale und/oder ökologische Ziele verfolge.

Maria Anastasiadis beschreibt die Vielfalt an Termini als ein Begriffsdickicht, in dem unterschiedliche Begrifflichkeiten synonym gebraucht werden, selbst wenn sie mitunter Verschiedenes benennen. Sie verortet die Solidarökonomie mit all ihren Ausprägungen im sogenannten „Dritten Sektor“ (vgl. Anastasiadis o.A.). Anastasiadis, Essl, Riesenfelder, Schmid und Wetzel haben in einer umfassenden Studie den Dritten Sektor in Wien untersucht und diesen Begriff als dritten Pol neben dem ersten Sektor, dem gewinnorientierten privaten, und dem zweiten Sektor, dem staatlich öffentlichen, definiert (vgl. Anastasiadis / Essl / Riesenfelder / Schmid / Wetzel 2003:11). Der Dritte Sektor ist von den beiden anderen weniger anhand der Frage, *welche* Produkte und Dienstleistungen hervorgebracht werden, zu unterscheiden, als vielmehr durch die Frage, *wie* die Produkte und Dienstleistungen hervorgebracht werden (vgl. ebd.:8). Sie werden weder nach kapitalistischen, marktwirtschaftlichen Regeln noch durch den Staat erzeugt. Initiativen und Projekte innerhalb des Dritten Sektors nehmen am Gemeinwohl orientierte Aufgaben wahr, können von politischen Utopien getragen sein und knüpfen mitunter an alltagspraktische, reproduktive Notwendigkeiten der Beteiligten an (vgl. ebd.:9).

Die Verortung der Solidarökonomie im Dritten Sektor zeigt erste Konturen auf, die für das Verständnis des Begriffes grundlegend sind. Für den Zweck dieser Forschung wird auf die folgende Definition von Embshoff und Giegold verwiesen:

„Solidarische Ökonomie bezeichnet Formen des Wirtschaftens, die menschliche Bedürfnisse auf der Basis freiwilliger Kooperation, Selbstorganisation und gegenseitiger Hilfe befriedigen." (Embshoff / Giegold 2008:11) Diese Definition entstand in Anlehnung an die Erklärung von Lima vom Internationalen Netzwerk zur Förderung der sozialen und solidarischen Ökonomie, Ripess. Solidarität bedeutet danach, die Bedürfnisse *aller Beteiligten* zu berücksichtigen und sich damit von den vorherrschenden Regeln der Marktwirtschaft abzuheben (vgl. ebd.:11-13).

2.1.2. Analog und digital

Da der Begriff der Digitalisierung ein wesentlicher in dieser Arbeit ist, soll geklärt werden, was dieser im technischen Sinn bedeutet und was der Unterschied zwischen analog und digital ist. Analog kommt aus dem Altgriechischen und bedeutet verhältnismäßig. Analog ist ein Signal dann, wenn es in seinem Wert und in seinem zeitlichen Verlauf stetig und damit stufenlos ist. Digital leitet sich vom lateinischen Wort „digitus" ab und bedeutet Finger. Ein digitales Signal ist im Gegensatz zum analogen wert- und zeitdiskret, das heißt es besitzt Abstufungen (vgl. Wöstenkühler 2016:11-14; Fricke 2013:1-2). Zur Veranschaulichung der Unterscheidung soll das Beispiel einer Analoguhr dienen: Umgangssprachlich wird darunter eine Zeigeruhr verstanden, die im technischen Sinn nicht per se analog ist. In vielen Fällen „springt" der Minutenzeiger zu genau jeder Minute um genau eine Sechzigstel-Umdrehung weiter. Das bedeutet, das angezeigte Signal verändert sich stufenförmig in gewissen Zeitabständen und ist somit wert- und zeitdiskret. Im technischen Sinn wäre die Zeigeruhr erst dann analog, wenn sich der Zeiger proportional und stetig mit der verstrichenen Zeit weiterbewegt – also zu jeder noch so kleinen Zeitänderung eine entsprechend kleine Bewegung erfährt.

Digitale Technologien, die unter dem Schlagwort „Digitalisierung" Anstoß für weitreichende, gesellschaftliche Veränderung waren und sind, operieren mit (elektrischen Spannungs-) Signalen, die wert- und zeitdiskret sind. Der Vorteil eines digitalen Signals ist jener, dass durch die Abstufung des Signals eine Unterscheidung zweier Werte vereinfacht wird (vgl. ebd.). Der Vorteil der Genauigkeit eines analogen Signals wird zugunsten einer besseren Unterscheidbarkeit aufgegeben. In binären digitalen Schaltungen, wie z.B. einem Computer, wird dies auf die radikalste Art und Weise umgesetzt. Es werden lediglich zwei Werte unterschieden: 0 oder 1; Ja oder Nein; Low oder High. Eine Größe, die nur zwei mögliche Informationen enthalten kann, nennt man Bit. Mit einer Aneinanderreihung mehrerer Bits kann die Zahl der darstellbaren Zustände

erhöht werden[2]. Damit digitale (binäre) Technologien Signale aus der „analogen Welt“ verarbeiten können, muss daher ein stetiges Signal zuerst in diskrete Werte abgestuft und als eine bestimmte Abfolge von Bits kodiert werden.

2.1.3. Software und Sourcecode

Bei digitalen Geräten kann grundsätzlich zwischen Software und Hardware unterschieden werden. Die Hardware bezeichnet alle physischen Komponenten eines Computers, also die elektrische Schaltung mit all ihren Bauteilen und die daraus resultierenden Funktionen (vgl. Gumm / Sommer 2011:35). Die Software bzw. das Programm ist eine Abfolge von Anweisungen, die die Hardware steuert. Bei der Programmierung von Software werden diese Steuerungsbefehle in Textform geschrieben. Für die Ausführung muss das Programm durch einen sogenannten Compiler in Maschinenbefehle übersetzt werden, in Bitfolgen, die von der Hardware in Form von elektrischen Signalen verarbeitet werden können (vgl. ebd.:16-17). Dem Begriff der Software kann man sich aus drei Perspektiven nähern: die erste, Software als Steuerung der Hardware, wurde soeben erläutert. Eine zweite Perspektive kann die der Endanwender*innen sein. Sie *sehen* die Software über die grafische Benutzerschnittstelle des Programms über ihren Bildschirm. In den meisten Fällen ist es das GUI (graphical user interface), das die Software und deren Funktionen für den Menschen sinnlich erfahrbar macht. Die dritte Sicht ist jene der Softwareentwicklung. Programmiert wird Software in Programmiersprachen, wie zum Beispiel C, Java, PHP, Python etc. Eine Programmiersprache hat, ähnlich zu andere Sprachen, ihre eigenen Worte (Befehle und Anweisungen) und ihre eigene Grammatik (Syntax). Der Sourcecode, bzw. auch Quelltext, Quellcode oder Programmcode genannt, ist das in einer Programmiersprache formulierte Schriftstück, welches die Software darstellt (vgl. Precht / Meier / Tremel 2004:538).

2.1.4. Open Source

Open Source (abgekürzt OS bzw. OSS für Open-Source-Software) kann als „quelloffen“ übersetzt werden. Das bedeutet, dass der Sourcecode eines Programms veröffentlicht wird und damit einsehbar ist. Damit ist zunächst noch nicht gesagt, wie das Programm entstanden ist, ob die Software gratis ist bzw. was mit dem offenen Sourcecode gemacht werden darf (vgl. Grassmuck 2004:231).

2 Ebenso funktioniert das Dezimalsystem. Ist eine Einerstelle unzureichend – es werden mehr als als zehn Zustände (0-9) benötigt – dann wird die Zahl um eine Zehnerstelle erweitert und es können somit 100 Zustände (0-99) dargestellt werden.

Der Begriff entwickelte sich aus jenem der „Free Software“ bzw. „Freie Software“. Diese Bezeichnung wurde stark durch die Person Richard Stallman geprägt und verteidigt. Mit „frei“ sei dabei nicht der Preis, sondern die Freiheit gemeint. Free Software muss folgende vier Grundfreiheiten einräumen: *erstens* den Zugang zum Sourcecode, *zweitens* die Rechte, die Software zu kopieren und weiterzugeben, *drittens* das Recht, die Software zu verändern, und *viertens,* sie unter denselben Bedingungen wieder zu veröffentlichen (vgl. Kharitoniouk / Stewin 2004:6). Der Begriff wurde Ende der 1990er Jahre von einigen Vertreter*innen dieser Softwarebewegung zunehmend durch jenen des „Open Source“ abgelöst (vgl. Grassmuck 2004:230).

Die Definition von der Open-Source-Initiative (OSI) beinhaltet zehn Merkmale von Open-Source-Software, die denen der Free Software zum Großteil ähnlich sind, sich jedoch in manchen Punkten unterscheiden. Diese wären, dass

- Kopien von Open-Source-Software auch verkauft werden dürfen. Jedoch darf dabei nur die Dienstleistung der Verbreitung verkauft werden und nicht die Software im Sinne einer Lizenzgebühr.
- Weiterentwicklungen von Open-Source-Software nicht zwingend die gleiche Lizenzierung wie das Ursprungsprogramm aufweisen müssen.
- keine bestimmte Nutzungsform verboten werden darf. Somit kann Open-Source-Software auch kommerzialisiert werden (vgl. OSI 2007; Kharitoniouk / Stewin 2004:7-9).

Die Lizenzen für Open-Source-Software sind somit liberaler als jene der Free Software. Stallman streicht jedoch auch einen Unterschied in den Wertvorstellungen hervor:

> „Open Source ist eine Entwicklungsmethode, bei Freier Software geht es um eine soziale Bewegung. Für die Freie-Software-Bewegung ist Freie Software ein ethischer Imperativ, denn nur Freie Software respektiert die Freiheit des Nutzers. Im Gegensatz dazu fragt die Open-Source-Philosophie immer nur danach, wie man Software besser machen kann, also nach praktischen Aspekten.“ (Stallman 2007:2)

Abseits von Ideologien stellen Kharitoniouk und Stewin fest, dass Free Software immer den Kriterien von Open-Source-Software entspreche, dieses Verhältnis umgekehrt jedoch nicht gelte. Open-Source-Software kann, im Gegensatz zur Free Software, unter anderen Nutzungsbedingungen veröffentlicht werden. Da damit die Kriterien ausgeweitet werden und die Quelloffenheit ohnehin beiden Begriffen gemein ist, hat sich jener der Open-Source-Software für freie und quelloffene Programme etabliert (vgl. Kharitoniouk / Stewin 2004:9). Der Begriff „Open

Source“, wie er in dieser Bachelorarbeit verwendet wird, umfasst gleichermaßen Free Software als auch Open-Source-Software gemäß der OSI-Definition. Eine analytische Trennung der beiden Begriffe wäre nicht zweckmäßig, da in dieser Arbeit der kollaborative Prozess der Softwareentwicklung im Vordergrund stehen soll und dieser beiden Begriffen gemein ist.

Um Open-Source-Software von anderen gebräuchlichen Begriffen abzugrenzen, soll folgende Tabelle von Kharitoniouk und Stewin dienen (2004:10-11):

„I. Der Quellcode ist modifizierbar.
II. Modifikationen müssen immer unter denselben Bedingungen veröffentlicht werden.
III. Das Produkt ist lizenzgebührenfrei.
IV. Die Nutzung ist uneingeschränkt.
V. Die Weiterverteilung ist erlaubt.

	Kurzbeschreibung	I	II	III	IV	V
Free Software	Der Quellcode ist offen und seine Modifikationen müssen auch offen bleiben.	X	X	X	X	X
Open-Source-Software	Quelloffene Software soll Unternehmen und Wirtschaft näher gebracht werden. Die kommerzielle Nutzung soll einfacher sein (im Vergleich zu Free Software).	X		X	X	X
Public Domain	Diese Software ist als ein Sonderfall zu betrachten: Der Urheber verzichtet komplett auf das Copyright. Somit wird diese Software zum Gemeingut und kann uneingeschränkt genutzt werden. Sollte der Quellcode zur Verfügung stehen, liegt Open-Source-Software vor.	X		X	X	X
Freeware	Diese Art der Software ist keine Free Software. Es werden zwar keine Lizenzgebühren erhoben, aber der Quellcode steht nicht zur Verfügung.			X	X	X
Shareware	Hierunter wird Software verstanden, die für eine vom Autor festgelegte Testphase genutzt werden darf. Ist die Testphase abgelaufen, so sind Lizenzgebühren zu bezahlen.					X
Übliche proprietäre Software	Proprietäre Software wird als „Gegenmodell“ zur Free bzw. Open-Source-Software gesehen: Die Weiterverteilung ist verboten, der Quellcode bleibt verborgen, sämtliche Modifikationen sind verboten und es werden immer Lizenzgebühren verlangt. Es bleiben fast alle Rechte beim Anbieter.“					

2.2. Das Open-Source-Konzept

Nach dem die wesentlichen Begriffe erörtert wurden, soll im Folgenden dargelegt werden, wie sich das Open-Source-Konzept entwickelte, auf welchen Prinzipien es fußt und was es bei den dazugehörigen Lizenzmodellen zu beachten gilt.

2.2.1. Die Entstehung

Proprietäre Software als Pendant zur Open-Source-Software legt den Sourcecode nicht offen (Closed Source). Die Endanwender*innen bekommen nur das fertige, ausführbare Programm, welches nicht einseh- oder veränderbar ist. Die Open-Source-Bewegung entstand, als die PC-Hardware standardisiert wurde und in die Privathaushalte Einzug hielt. Mit dem Vorhandensein der Hardware wurde dementsprechend die Nachfrage an Software größer. Unter den Endanwender*innen entstanden Programmierer*innen, die Software nicht als Blackbox ansehen wollten, sondern die Funktionsweise verstehen und Programme weiterentwickeln wollten. Mit dem Aufkommen des Internets – dessen Entwicklung anhand der offenen Standards und der Partizipation der Community in der Anfangsphase dem Open-Source-Konzept ähnlich war – wurde eine Entwicklungsumgebung für die Open-Source-Community geschaffen. Durch das Medium „Internet“ wurde Open-Source-Software verteilt, es wurde darüber kommuniziert und in Kooperationen daran programmiert. Eine der bedeutendsten, wenn auch damals noch nicht als Open Source bezeichnete, Entwicklungen war das GNU/Linux Betriebssystem, das bis heute weiterentwickelt wird und vor allem im Serverbereich große Anwendung findet (vgl. Kharitoniouk / Stewin 2004:3-5).

2.2.2. Die Kathedrale und der Basar

1999 verfasste der Softwareentwickler Eric S. Raymond das Essay „Die Kathedrale und der Basar“ über das Programmieren von Open-Source-Software. Ausgehend vom großen Erfolg des GNU/Linux Projektes leitete er Grundsätze der Open-Source-Softwareentwicklung ab und überprüfte sie anhand seines eigenen Open-Source-Projektes „fetchmail“. Er unterscheidet zwei Modelle der Softwareentwicklung: die „Kathedrale“ und den „Basar“. Proprietäre Unternehmen haben ein abgeschlossenes Entwicklerteam, eine „Elite“, die alleinig an einem Softwareprojekt arbeitet. Dies gleicht dem Bau einer Kathedrale, in der einige wenige eingeweihte Spezialist*innen an der Umsetzung beteiligt sind. Demgegenüber gleicht die Open-Source-Community einem Basar, der sich selbst organisiert, veränderbar und für alle zugänglich ist (vgl. Raymond 1999).

Raymond leitete 19 Thesen ab, von denen einige, die die Prinzipien in einem Open-Source-Projekt veranschaulichen, dargelegt werden sollen:

- „Jede gute Software beginnt mit den persönlichen Sehnsüchten eines Entwicklers." (ebd.:4)
- „Gute Programmierer wissen, welchen Code sie schreiben sollen. Großartige Programmierer wissen, welchen Code sie umschreiben (und recyceln) können." (ebd.:4)
- „Sobald man das Interesse an seinem Programm verliert, ist es die letzte Pflicht, es einem kompetenten Nachfolger zu überlassen." (ebd.:6)
- „Die Anwender als Mit-Entwickler zu sehen ist der Weg zu schnellen Verbesserungen und Fehlerbehebungen, der die geringsten Umstände macht." (ebd.:7)
- „Wenn man einen ausreichend großen Stamm an […] Mit-Entwicklern hat, wird jedes Problem schnell identifiziert und die Lösung jedem offensichtlich sein." (ebd.:9)

Nach Raymond kann das Konzept von Open-Source-Softwareentwicklung folgendermaßen zusammengefasst werden: Ein Projekt beginnt mit einem Problem, für das eine oder mehrere Personen eine Softwarelösung programmieren. Dabei können sie sich anderer quelloffener Codes bedienen. Der neu entstehende Sourcecode wird jedoch nicht geheim gehalten, sondern wird, wie auf einem Basar, für alle Anwender*innen und andere Interessierte offengelegt. So kann sich um das Projekt eine Interessengruppe, die *Community*, bilden. Diese tritt in ein Austauschverhältnis mit den Urheber*innen der Software, indem sie einerseits selbst vom Programm profitiert und andererseits durch Partizipation in der Mitentwicklung dazu beiträgt, dass auch die Urheber*innen unterstützt werden. Das somit wachsende Team wird zu einer wertvollen Ressource für das Projekt. Die „Gemeinde" wird zum „Talentepool" (vgl. Smid 2006:22).

2.2.3. Open-Source-Lizenzen

Die Lizenzierung von Software dient dazu, Rechte und Pflichten im Zusammenhang mit deren Verbreitung, Verwendung und Entwicklung zu regeln. Im Open-Source-Bereich gibt es eine Vielzahl verschiedener Lizenzmodelle – weit verbreitet sind beispielsweise Apache-2.0, BSD-2 und 3, (L)GPL, MPL2.0 (vgl. OSI o.A.). Widmer und Bähler weisen darauf hin, dass die verschiedenen Lizenzen zwar die Grundprinzipien des Open-Source-Konzeptes gemein haben, sich jedoch vor allem für den Vertrieb und die Weiterentwicklung Unterschiede ergeben. Diese können die Bekanntgabe der Autor*innen, das Sichtbarmachen von Veränderungen oder die Form der Weitergabe des Sourcecodes betreffen (vgl. Widmer / Bähler 2006:167-168).

> „Das wichtigste Kriterium zur Beurteilung der verschiedenen OSS-Lizenzen ist, ob diese ein Copyleft enthalten oder nicht. […] Lizenzen, die ein Copyleft beinhalten, verlangen, dass Software, die durch Modifikation der unter der Lizenz erworbenen OSS oder durch die Übernahme von Teilen der OSS erstellt wurde, ebenfalls wieder unter der gleichen Lizenz als OSS zur Verfügung gestellt wird. Das Konzept des Copyleft ist für OSS nicht zwingend. Es liegt jedoch vielen der für OSS verwendeten Lizenzen zugrunde, insbesondere der am meisten verbreiteten GPL.“ (ebd.:169)

Für die Endanwender*innen von Open-Source-Software haben die Unterschiede der verschiedenen Lizenzen wenig Bedeutung, da die Rechte zur freien Nutzung und Vervielfältigung weitestgehend in allen Modellen eingeräumt werden (vgl. ebd.:172). Demgegenüber gilt es für Entwickler*innen zu beachten, ob in der Lizenzierung des verwendeten Codes ein Copyleft vorgesehen wird und in welchem Umfang (vgl. ebd.:175).

2.3. Verortung von Open Source in der Solidarökonomie

Folgt man der Definition, dass Solidarökonomie eine Form des Wirtschaftens sei, die menschliche Bedürfnisse auf der Basis freiwilliger Kooperation, Selbstorganisation und gegenseitiger Hilfe befriedigt, dann wird anhand der bisherigen Darlegungen eine Parallele zwischen dem Open-Source-Konzept und jenem der Solidarökonomie ersichtlich. Open-Source-Software wird in der Regel von freiwilligen Mitarbeiter*innen, die dafür nicht monetär entlohnt werden, in gemeinsamer Zusammenarbeit und Kooperation organisiert und programmiert. Unter den Initiator*innen und den später folgenden Mitentwickler*innen herrscht ein reziprokes Verhältnis, da die Partizipation einer Person für alle Beteiligten hilfreich sein kann. Verbesserungen in der Software haben gleichzeitig einen Eigen- und einen Fremdzweck. Open-Source-Projekte sind meist, ähnlich zu solidarökonomischen Initiativen, nicht nur Bottom-Up-, sondern auch im weitesten Sinne dezentrale Prozesse. Die Entwickler*innen sind mitunter über Staatsgrenzen hinweg räumlich voneinander getrennt.

Die entstehenden Produkte der Open-Source-Softwareentwicklung können, bezugnehmend auf das folgende Zitat, als Commons angesehen werden.

> „Commons sind mithin keine Güter, wenngleich sie oft als solche beschrieben werden. Denn Güter sind nicht aufgrund ihrer »natürlichen« Eigenschaften Commons, sondern sie müssen erst dazu gemacht werden. Commons lassen sich im Wesentlichen als institutionelles, rechtliches und infrastrukturelles Arrangement für ein Miteinander – das Commoning – beschreiben, bei dem Nutzung, Erhaltung und Produktion vielgestaltiger Ressourcen gemeinsam organisiert und verantwortet werden. Die Regeln des Commoning werden (idealerweise) im gleichberechtigten Miteinander von Peers festgelegt, deren Bedürfnisse im Mittelpunkt stehen.“ (Acksel / Euler / Gauditz / Helfrich / Kratzwald / Meretz / Stein / Tuschen 2015:134).

Grundsätzlich können verschiedenste Ressourcen als Commons organisiert sein. Das Spektrum reicht von solchen zur Deckung menschlicher Grundbedürfnisse, wie zum Beispiel Wasser, bis hin zu immateriellen Gütern, wie Open-Source-Software. Das Zurverfügungstellen von Commons kann eine Strategie der Armutsbekämpfung sein, die laut Helfrich eine, im Gegensatz zu fiskal- und sozialpolitischen Interventionen, primäre Umverteilung der Ressourcen bewirke (vgl. Helfrich 2013). Die Grundprinzipien des Open-Source-Konzeptes, wie beispielsweise der freie Zugang zur Software, die Möglichkeiten der Mit- und Weiterentwicklung und der Anpassung der Programme an die eigenen Bedürfnisse, bilden das Nutzungsarrangement, welches sich als Commoning zusammenfassen lässt und somit die Open-Source-Software in Form von Commons der Community zur Verfügung gestellt wird.

2.4. Digitalisierung in der Sozialen Arbeit

Nachdem die Verbindung zwischen Open Source und Solidarökonomie aufgezeigt wurde, soll an nächster Stelle die Bedeutung der Digitalisierung für die Soziale Arbeit, wenn auch nur partikular, erörtert werden, um einen Einstieg in diese Thematik zu bieten und um an die Forschungsfrage anknüpfen zu können.

Der gesellschaftliche Prozess der Digitalisierung bringt einerseits Chancen, wie die Vernetzung verschiedenster Personen und Gruppen oder wie das Speichern und Verbreiten von digitalen Commons (zum Beispiel Wissen oder Open-Source-Programmen). Andererseits können ein und dieselben Möglichkeiten auch zum Risiko werden, indem problematische Informationen verbreitet oder persönliche Daten von Dritten gespeichert und verwertet werden (vgl. Steiner 2017). Steiner fährt in seinem Artikel fort, dass Chancen wie auch Risiken ein unauflösbares Dilemma darstellen, und zeigt diese, in Anlehnung an Friedrich Krotz (vgl. 2001), anhand dreier Dimensionen – Zeit, Raum und Soziales – auf:

- Zeitlich: Digitale Medien ermöglichen die mobile und ununterbrochene Erreichbarkeit. Neben den Chancen der besseren Organisation verschiedenster Prozesse birgt dies ebenso den Druck des „always on“ auf die Teilnehmer*innen.
- Räumlich: Der virtuelle Raum kann dazu dienen, geografische Grenzen zu überwinden, oder auch dazu beitragen, öffentlichen Raum zu organisieren. Demgegenüber kann es dadurch auch zur Isolation im physischen Raum kommen („Bedroom Cultures“).
- Sozial: Während einerseits Möglichkeiten der Partizipation in digitalen Medien erschlossen

werden, können andererseits die Privatsphäre bedroht und soziale Ungleichverhältnisse im Internet reproduziert werden („Digital Divide“) (vgl. ebd.).

Digitale Medien kann man nicht per se einer Bewertung unterziehen - sie sind Mittel zum Zweck, unabhängig davon, was dieser sein mag. Sie können gleichermaßen für konstruktives wie auch für destruktives Verhalten benutzt werden. Selbst der an sich positive Zweck kann negative Auswirkungen haben, wenn man beispielsweise bei Social Media an die Möglichkeit der Vernetzung mit dem Umfeld und dem gleichzeitigen Suchtpotential denkt.

In der Sozialen Arbeit wird man sich, insoweit man das nicht bereits macht, mit der Digitalisierung auseinandersetzen müssen, da diese Teil der Lebenswelten der Klient*innen ist und andererseits die Profession selbst Teil einer digitalen Gesellschaft ist. Eike Rösch macht darauf aufmerksam, dass der Ansatz der Lebensweltorientierung auch die digitalen Tools, die von Klient*innen benutzt werden, umfassen sollte. Einige der verwendeten Medien weisen asymmetrische Machtverhältnisse auf, in denen die Nutzer*innen nicht mehr über ihre Daten verfügen können und somit ihre Handlungs- und Entscheidungsfreiräume eingeschränkt sind. Er weist dabei nur allgemein auf die Potentiale von Open-Source-Tools hin (vgl. Rösch 2018).

Um diese Potentiale von Open-Source-Software für die Soziale Arbeit – nicht nur im Sinne von asymmetrischen Machtverhältnissen, sondern allgemein – auszuloten, führte ich drei Experteninterviews, die im folgenden Kapitel behandelt werden.

3. Forschungsdesign

Aus der Literaturrecherche wurde ersichtlich, dass bislang Open-Source-Software(-entwicklung) und das dahinterstehende Konzept in Anbetracht auf die Potentiale für die Soziale Arbeit – bis auf den zuletzt zitierten Eike Rösch – kaum wissenschaftlich behandelt wurden. Darüber hinaus waren ebenso wenig Institutionen oder Sozialarbeiter*innen zu finden, die Open Source im Zusammenhang mit ihrer Tätigkeit thematisiert hätten[3]. Darum galt für diese Forschung, ein noch wenig beleuchtetes Feld zu erkunden.

3 Demgegenüber wurde Open Source aus verschiedensten anderen Blickwinkeln bereits mehrfach wissenschaftlich aufgearbeitet – beispielsweise in Hinblick auf ökonomische, innovative, technische, soziologische oder gruppendynamische Aspekte.

3.1. Forschungsfrage

Aufgrund der fehlenden theoretischen wie auch praktischen Bezüge war der Anspruch dieser Forschung eine erste Exploration des Feldes. Die Hauptfrage wurde so formuliert, dass sie das Forschungsinteresse abgrenzt und gleichzeitig so weit offen ist, dass verschiedenste Aspekte aufgenommen werden können. Die **Hauptfrage** lautet:

„Welche Nutzungsmöglichkeiten für die Soziale Arbeit ergeben sich aus der Open-Source-Softwareentwicklung?“

Aus den theoretischen Überlegungen und vor allem während des Forschungsprozesses ergaben sich Schwerpunkte, die in folgenden **Unterfragen** repräsentiert werden sollen:

- *„Wie wird die solidarökonomische Entwicklung von Open-Source-Software organisiert?“*
- *„In welcher Hinsicht kann Open-Source-Software für Klient*innen, Sozialarbeiter*innen und Organisationen nützlich sein?“*
- *„Wie könnten Sozialarbeiter*innen in der Softwareentwicklung partizipieren?“*

3.2. Forschungsablauf und Erhebungsinstrumente

Nach der Literaturrecherche war ersichtlich, dass es zum einen an spezifischen Informationen fehlte und zum anderen ein disziplinübergreifender Bezug zwischen Sozialer Arbeit und der Softwareentwicklung hergestellt werden musste. Aufgrund dessen führte ich drei Interviews mit Experten aus der Informationstechnik wie auch der Sozialen Arbeit.

Das erste Interview war ein narratives, welches mit einem Hochschuldozenten für Informatik geführt wurde. Dieser ist selbst Mitentwickler in Open-Source-Projekten. Dieses Interview diente dazu, aus Sicht eines Praktikers einen ersten Überblick über die Open-Source-Entwicklung und die Community zu bekommen. Das zweite Interview wurde mit dem Sozialarbeits-Professor und Institutsleiter des Ilse Arlt Instituts für Soziale Inklusionsforschung FH-Prof. Mag. Dr. Johannes Pflegerl geführt, der auf Basis seiner sozialwissenschaftlichen Forschung, in Kooperation mit einem technischen Studiengang, ein digitales Software-Tool entwickelt hat. Anhand eines Leitfadeninterviews wurden Themen wie die Digitalisierung und die Softwareentwicklung in der Sozialen Arbeit besprochen. Ein drittes Experteninterview wurde ebenfalls mit Leitfaden mit einem Unternehmer der IT-Branche geführt, der selbst Open-Source-Software mitentwickelt. Dabei wurden vor allem Vor- und Nachteile von Open-Source-Software im Vergleich zu proprietärer

Software, verschiedene Anwendungsbereiche und die Partizipationsmöglichkeiten der Sozialen Arbeit in der Softwareentwicklung behandelt. Die Interviews fanden jeweils im Büro der interviewten Person statt und dauerten zwischen 45 Min. und einer Stunde. Zur Auswertung wurden die Interviews transkribiert.

3.3. Qualitative Inhaltsanalyse

Für die Auswertungsmethode wurde von einem rekonstruktiven Verfahren abgesehen, weil das Interesse weniger bei dem Herausarbeiten von latenten Sinnstrukturen lag, als vielmehr beim Ordnen und Kategorisieren der Daten der Experteninterviews. Dafür wurde die qualitative Inhaltsanalyse nach Philipp Mayring angewandt. Die verwendete Technik nennt sich „Zusammenfassung und induktive Kategorienbildung“. Dabei wird versucht, alles Datenmaterial zu berücksichtigen und zusammenzufassen. Die Kategorisierung der Daten erfolgt aus dem Datenmaterial heraus und nicht anhand vorheriger Überlegungen (Induktion) (vgl. Mayring 2015:68).

Der erste Schritt in diesem Verfahren ist die Bestimmung des Ausgangsmaterials (vgl. ebd.:54-57). In diesem Fall sind es die drei Interviewtranskripte, die wie im vorherigen Unterkapitel beschrieben zustande gekommen sind. Der nächste Schritt ist, die Fragestellung für die Analyse festzulegen (vgl. ebd.:58-60). Hier lässt sich auf die Forschungsfrage samt ihren Unterfragen verweisen. Als nächstes wird die Kodiereinheit festgelegt, also der kleinste auszuwertende Teil des Datenmaterials (vgl. ebd.:61-64). Diese wurde für jede vollständige Aussage einer der interviewten Personen über den jeweils befragten Gegenstand festgelegt.

Das Datenmaterial wird ausgewertet, indem das Transkript in die Kodiereinheiten unterteilt und jede Kodiereinheit hinsichtlich der wesentlichen Aussage paraphrasiert wird. Danach wird das Abstraktionsniveau der Paraphrasen bestimmt. Dabei wird festgelegt, inwieweit eine Aussage über einen Sachverhalt vom spezifischen Fall losgelöst und verallgemeinert wird. All jene Paraphrasen, die unter dem Abstraktionsniveau liegen, werden weiter verallgemeinert. Paraphrasen, die ein höheres Abstraktionsniveau aufweisen, werden belassen. Im nächsten Schritt werden Paraphrasen, die sich wiederholen oder bezüglich der Fragestellung keinen Sinninhalt aufweisen, gestrichen. Danach werden alle Paraphrasen, die sich aufeinander beziehen oder in Zusammenhang stehen, zu einer neuen Aussage gebündelt. Diese neuen Aussagen bilden bereits ein Kategoriensystem. Dabei kann rücküberprüft werden, ob alle Kodiereinheiten im Kategoriensystem repräsentiert werden. Sollte das Kategoriensystem noch zu umfangreich sein,

dann können die Aussagen ein weiteres Mal abstrahiert werden (vgl. ebd.:69-84). Da die Interviews jeweils abgegrenzte Themenbereiche behandelten, wurde für jedes Interview ein eigenes Kategoriensystem ausgearbeitet und danach zu einem umfangreichen gebündelt.

4. Ergebnisse

Aus der Auswertung der Interviews gingen drei Themenschwerpunkte hervor, die durch die nachfolgenden Unterkapitel repräsentiert werden. Aussagen der Experten, die sinngemäß wiedergegeben werden, sind im Konjunktiv formuliert. Zur Abgrenzung davon sind eigene Interpretationen und weiterführende Gedanken im Indikativ geschrieben. In diesem Kapitel werden die Ergebnisse allgemein dargelegt. Im darauffolgenden Resümee werden diese als Antwort auf die Forschungsfrage zusammengefasst und interpretiert.

4.1. Ablauf von Open-Source-Projekten

Um Open-Source-Softwareentwicklung in Hinblick auf die Nutzungsmöglichkeiten für die Soziale Arbeit zu untersuchen, sollen auch die Motivation der Entwickler*innen und Gründer*innen, die Organisations- und Arbeitsweise und die entstehenden wie auch benötigten Ressourcen in Open-Source-Projekten beleuchtet werden. Die Daten für diese Ergebnisse stammen vor allem aus den Interviews mit dem IT-Dozenten und dem IT-Unternehmer, die beide an Open-Source-Projekten beteiligt waren.

4.1.1. Organisation und Arbeitsweise

Das Konzept von Open Source schreibe grundsätzlich keine Organisationsform vor. Projekte könnten demnach hierarchisch strukturiert sein und auch autoritär geführt werden. Demgegenüber können sie ebenso eine flache Hierarchie aufweisen und basisdemokratisch organisiert sein. Unabhängig von der Organisationsform stellen Open-Source-Projekte zielgerichtete Strukturen dar, deren rechtlicher Rahmen, wie z.B. Urheberrechtsfragen, durch Lizenzmodelle geklärt wird. Bei größeren Projekten sei die Gründung eines Vereins oder einer anderen Rechtsform nicht unüblich. Entscheidend für die Organisationsweise sei vor allem die Person bzw. seien die Personen, die das Projekt gründeten – sogenannte „Maintainer“[4]. Maintainer haben die Zugangsrechte zum Ursprungsprojekt und sind somit die entscheidende Instanz über

4 „to maintain“ kann mit dem Wort „aufrechterhalten“ übersetzt werden. Mit Maintainer wird der/die Hauptentwickler*in bzw. Gründer*in in einem Softwareprojekt bezeichnet.

Veränderungen im Sourcecode. Nutzer*innen der Software oder auch Interessierte können den Quellcode zwar einsehen, kopieren, für sich selbst verändern und diese Veränderungen in das Originalprojekt einreichen, eine tatsächliche Veränderung des ursprünglichen Programms bleibt jedoch den Maintainern vorbehalten. Auf das Projekt bezogen entsteht dadurch eine Machtasymmetrie zwischen Maintainern und Mitentwickler*innen.

Für das Projekt könne auch ein Kernteam aufgebaut werden. Dieses grenze sich von anderen Mitentwickler*innen dahingehend ab, dass es eine kleinere, abgeschlossene Gruppe ist, die durch eigene, für die restliche Community nicht sichtbare, Kanäle kommuniziert, intensiver am Projekt arbeitet und mehr Einfluss darauf ausübt. Bekommen die Entwickler*innen im Kernteam die Zugangsrechte, werden sie zu Maintainern. Dies wäre jedoch nicht zwingend notwendig. Mitentwickler*in an einem Projekt sei man unabhängig vom geleisteten Beitrag. Hätte eine Person zu einem Programm nur eine einzige Programmzeile beigetragen, wäre sie ebenso Mitentwickler*in. Demgegenüber würden Personen dann in das Kernteam aufgenommen werden, wenn sie engagiert sind und viel zum Projekt beitragen. Maintainer besäßen lediglich die Zugangsrechte zum Projekt. So könne es vorkommen, dass Entwickler*innen aus dem Kernteam ohne Zugangsrechte mehr zum Projekt beitragen, als der/die Urheber*in. Maintainer hätten jedoch über die Programmierung hinausreichende Aufgaben. Sie sollten beispielsweise die Visionen des Projektes aufrechterhalten und die Community zusammenhalten. Da bei diesen Personen die Entscheidungskompetenz liege, müssen bei ihnen auch mögliche Diskrepanzen innerhalb der Community oder des Entwickler*innen-Teams gelöst werden. Maintainern werde nicht nur Führungskompetenz abverlangt, sondern sie sollten auch das Projekt nach außen hin vertreten.

Open Source ermöglicht nicht nur, sich an einem Projekt zu beteiligen, sondern auch, den Sourcecode zu kopieren und auf dieser Softwarebasis ein neues Projekt zu beginnen. Diesen Vorgang nennt man „Fork“ (englische Bezeichnung für „Gabelung“). Ein möglicher Grund für einen Fork könne einerseits sein, dass es Meinungsdifferenzen bzw. unterschiedliche Zielvorstellungen im Originalprojekt gibt und daher ein neues begonnen wird. Andererseits könne es sein, dass eine Open-Source-Software nicht weiterentwickelt wird und das Projekt seitens des/der Maintainer nicht an andere Entwickler*innen abgegeben wird. Forks weisen sowohl Vor- als auch Nachteile auf, die es abzuwägen gilt. Ein Vorteil sei, dass im Gegensatz zur „analogen Welt“ das Endprodukt beliebig duplizierbar ist. Eine Kopie des Projektes ist mit keinen Kosten verbunden und erzeugt in dem Sinne keinen Schaden, wenn es sich nicht etabliert. Ein Nachteil entstehe jedoch hinsichtlich der Community. Zur Weiterentwicklung einer Open-Source-Software braucht es die Entwickler*innen

und Nutzer*innen, die Menschen, die sich am Projekt beteiligen. Mit einem Fork könne eine Teilung der Community einhergehen, die wiederum der Entwicklung und Innovation beider Projekte schaden könnte. Die Etablierung eines Forks hängt davon ab, inwiefern das angestrebte Ziel den Bedürfnissen der Community entspricht. Demgegenüber könne es im Interesse der Entwickler*innen liegen, anstatt für eigene Zwecke ein Projekt zu forken, die benötigten Veränderungen im Originalprojekt einzureichen. Denn damit werden die benötigten Funktionalitäten in die Originalsoftware eingepflegt. Spätere Updates des Originalprogramms enthalten dann bereits die Veränderungen und müssen nicht erneut implementiert werden.

4.1.2. Motivation zur Gründung und zur Partizipation

Ein Open-Source-Projekt könne grundsätzlich jede Person gründen. Aus den Interviews konnten zwei verschiedene Motive zur Gründung abgeleitet werden. Einerseits könne ein*e Entwickler*in altruistisch motiviert sein und eine Software, die er/sie für den Eigenbedarf programmiert hatte, als Open-Source-Projekt veröffentlichen, damit auch andere davon profitieren können. Das ursprüngliche Projekt müsse dabei nicht weiterentwickelt werden, sondern könne direkt genutzt werden oder als Grundlage für andere Projekte dienen. Dem zweiten Motiv liegt das Prinzip der Solidarökonomie zu Grunde. Eine Software werde demnach als Open-Source-Projekt veröffentlicht, um eine Community aufzubauen, in der Nutzer*innen auch als Mitentwickler*innen partizipieren. Die Softwareentwicklung des somit entstehenden, größeren Teams verhelfe zu einer Verbesserung des Programms, von der alle Beteiligten profitieren können. Dabei wird versucht, die Grenze zwischen Entwickler*innen und Nutzer*innen, wie sie bei proprietärer Software oder Freeware vorzufinden ist, durch reziproke Arbeitsverhältnisse aufzuheben. Das Vernetzungsmedium stelle das Internet dar, welches die Reichweite des Projekts bzw. der Ressourcenmobilisierung auf die gesamte Welt ausdehnt. Der Zugang zu den Projekten werde über Internetplattformen hergestellt, wie sie in Kapitel 4.2.2 näher beschrieben werden.

Eine der interviewten Personen berichtete, dass es für sie motivierend war, als ihre erste Einreichung einer Veränderung in einem Open-Source-Projekt aufgenommen wurde. Ihre eigene Weiterentwicklung am Sourcecode wurde in die Originalsoftware integriert. Dieser Vorgang wird als „Merge“ („to merge“ bedeutet übersetzt „fusionieren“) bezeichnet. Das Engagement wurde mit Einfluss im Projekt belohnt und dadurch wäre weiterer Antrieb zur Beteiligung geschaffen worden. Die Freiwilligkeit der Mitarbeit habe jedoch zur Konsequenz, dass sich im Projekt nur begrenzt Zwang ausüben lasse, da sonst die Partizipation verhindert werde. Druck müsse extrinsisch

belohnt werden – z.B. mit einer gewissen Position im Projekt. Werden Mitentwickler*innen stärker in das Projekt eingebunden, so entstehe auch ein stärkeres Gefühl der Verpflichtung. Dennoch komme es auch ohne Belohnungen dazu, dass gewisse Zwänge und Regeln geachtet und eingehalten werden, wenn sie dem Projekt dienen. So sei die Dokumentation bzw. die einheitliche Formatierung des Sourcecodes eine wenig gemochte Aufgabe, die trotz Freiwilligkeit in vielen Open-Source-Projekten umgesetzt werde. Des Weiteren würden Veränderungseinreichungen von Mitentwickler*innen nicht nur dankbar entgegengenommen werden, sondern kritisch geprüft und bei Bedarf Überarbeitung gefordert werden.

Einer der Experten erzählte von einem seiner Open-Source-Projekte, welches er und sein Kernteam mehrmals zu kommerzialisieren versucht hatten. Dies gelang ihnen jedoch nicht. Wie er meinte nicht deshalb, weil das Projekt nicht das Potential dafür hatte, sondern vielmehr deswegen, weil niemand der Entwickler*innen das Marketing, die Buchhaltung oder das finanzielle Risiko übernehmen wollte. Anstatt aus Zwang – wenn auch selbst auferlegt – solchen Aufgaben nachzugehen, wollten sie ihrer ursprünglichen freiwilligen Arbeit nachgehen und Entwickeln und Programmieren.

Unabhängig von der Motivation entstehe ein Open-Source-Projekt meist aus einer ersten Softwarebasis – aus einem Programm, das zwar nicht komplex oder fehlerfrei sein müsse, aber einen Grundstock für die Umsetzung einer Idee bildet – die der Maintainer zur Verfügung stellt. Demnach braucht es zur Bildung einer Community eine bereits erste Umsetzung der Software. Das spätere Entstehen einer Nutzer*innen- und Entwickler*innen-Gemeinschaft, die Feedback gibt und Veränderungen einbringt, bewirke, dass solche Projekte auch immer Kompetenzzentren für ihren definierten Zweck sind.

4.1.3. Ressourcen und Finanzierung

Mit der Größe der Community eines Open-Source-Projekts würden ebenso die Ressourcen im Projekt wachsen, da das Produkt nicht bloß mehr genutzt werde, sondern ebenso mehr Personen in der Softwareentwicklung partizipieren würden. Das Internet biete die Möglichkeit einer weltweiten Beteiligung an einem Projekt, woraus Diversität innerhalb der Entwickler*innen-Gemeinde resultiere. Neben Privatpersonen, die aus Eigeninteresse an Projekten partizipieren, könnten Softwareentwickler*innen auch von Firmen dafür beauftragt werden, an einem Projekt mitzuwirken. Das werde vor allem dann zum Fall, wenn ein Unternehmen von einem Open-Source-Projekt profitiert und beispielsweise die Bezahlung des/der Programmierer*in eine

geringere finanzielle Belastung darstellt als die Bezahlung von Lizenzgebühren alternativer, proprietärer Software. Falls ein Open-Source-Projekt über finanzielle Mittel verfügen sollte, dann wären diese hauptsächlich durch Spenden oder auch durch Förderungen eingenommen worden. Diese können den Entwickler*innen in Form von Sachleistungen zu Gute kommen, wie zum Beispiel Reisekosten für Team-Treffen, PCs, Arbeitsplätze etc.

4.2. Open-Source-Software und ihre Anwendung

Nachdem dargelegt wurde, wie Open-Source-Projekte entstehen, sollen Anwendungsbereiche aufgezeigt werden und weiterführend erörtert werden, wie Open-Source-Software(-Projekte) zu finden sind. Hierfür beziehe ich mich vor allem auf das Experteninterview des IT-Unternehmers.

4.2.1. Anwendungsbereiche

Dem IT-Unternehmer zu Folge gebe es Open-Source-Software für verschiedenste digitale Geräte wie PC, Handy oder Server. Grundsätzlich könne eine Unterscheidung von Support- und Wirkapplikationen getroffen werden. Während letzterer Begriff Software meint, die unmittelbar von den Nutzer*innen angewandt wird (z.B. ein Internet-Browser), ist mit ersterem Software gemeint, die für die Nutzer*innen unscheinbar arbeitet (z.B. Webserver, Datenbanken). Die Durchdringung von Open-Source-Software sei bei den Supportapplikationen höher als bei den Wirkapplikationen. Somit ist die Nutzung von Open-Source-Software oftmals indirekt und den Anwender*innen nicht bewusst – so z.B. die Verwendung von quelloffenen Apache-Webservern, wenn man im Internet surft.

Entscheiden sich Endanwender*innen für ein Open-Source-Betriebssystem bzw. -Programm, so würden die fehlenden Kosten häufig das Hauptargument sein. Speziell wenn Organisationen eine EDV-Landschaft aufbauen, können die Lizenzgebühren für proprietäre Betriebssysteme bzw. Anwendungssoftware beachtliche finanzielle Belastungen darstellen. Beim Einsatz von Open-Source-Software würden Kosten lediglich für den Support und gegebenenfalls für Adaptionen der Software anfallen. Neben dem finanziellen Aspekt bestehe noch jener des eigentlichen Grundgedankens von Open Source, der jedoch oftmals unberücksichtigt bleibe: Der Sourcecode ist offen und kann adaptiert werden. So könne verwendete Software auf die eigenen Bedürfnisse angepasst werden – z.B. eine Dokumentationssoftware für die spezifischen Anforderungen einer Organisation. Hier werde auch die Unterscheidung zur Freeware deutlich, da bei dieser der Quellcode geheim gehalten wird (= Closed Source).

4.2.2. Open-Source-Software finden

Zur Suche von Softwarelösungen aus dem Open-Source-Bereich gibt es verschiedene Plattformen im Internet. Im Interview wurden die folgenden drei Hostingportale genannt, die Möglichkeiten bieten, Open-Source-Software zu präsentieren, zu verwalten und herunterzuladen:

- SourceForge
- GitHub
- Google Code

SourceForge und GitHub seien die größten und bedeutendsten Portale. Bei Google Code solle darauf geachtet werden, dass Suchergebnisse kommerziell gefiltert sind – im Gegensatz zu SourceForge und GitHub. Die Suche nach Software erfolge am besten durch funktionelle Schlagwörter, die die Funktionen der gesuchten Software beschreiben. Open-Source-Projekte seien in vielen Fällen gut dokumentiert und in englischer Sprache beschrieben. Ein Merkmal zur qualitativen Beurteilung eines Projekts sei die Aktivität. Liegt die letzte Veränderung am Sourcecode – diese Daten sind in den Hostingportalen ersichtlich – bereits längere Zeit zurück, so sei dies ein Indikator, dass das Projekt nicht mehr gewartet wird. Für den sozialarbeiterischen Kontext sollte bei der Suche von Software, die Klient*innen-Informationen auf irgendeine Art und Weise verarbeiten soll, darauf geachtet werden, dass gesetzliche Anforderungen, wie z.B. Datenschutzrichtlinien, eingehalten werden können.

4.3. Soziale Arbeit in der Entwicklung von digitalen Tools

FH-Prof. Mag. Dr. Pflegerl berichtete im Interview, wie man am Studiengang Soziale Arbeit an der FH St. Pölten und am Ilse Arlt Forschungsinstitut mit der Digitalisierung umgehe und erläuterte, wie digitale Tools bereits entwickelt wurden und welche Herausforderungen damit verbunden seien. Mit den Ergebnissen soll sichtbar gemacht werden, welche Anforderungen es in der Sozialen Arbeit im Rahmen der Forschung an die Softwareentwicklung geben könnte und bereits gab, um in späterer Folge herausarbeiten zu können, welche Möglichkeiten diesbezüglich Open-Source-Ansätze bieten würden.

Hr. Pflegerl betonte, dass es in der Sozialen Arbeit allgemein, aber auch am Forschungsinstitut bzw. für die Lehre eine tiefere Auseinandersetzung mit dem Thema Digitalisierung brauche. Diese würde zukünftig forciert werden, um dieses gesellschaftliche Phänomen mit all seinen

Auswirkungen für den Kontext der Sozialen Arbeit einordnen und daraus Maßnahmen ergreifen zu können, so zum Beispiel die Beantwortung der Frage, ob die Digitalisierung in der Lehre vertieft werden solle. Auf Forschungsebene gebe es in der Sozialen Arbeit noch wenig Projekte, die Aspekte der Digitalisierung behandeln. Im Bereich der technischen Möglichkeiten sei Open Source daher ebenso ein neuer Gedanke, der noch nicht abgehandelt wurde. Nichtsdestotrotz wurden am Ilse Arlt Institut bereits drei digitale Tools im Zuge von Forschungsarbeiten entwickelt: Brelomate – eine Softwarelösung zur Vernetzung älterer, sozial isolierter Menschen (vgl. Ilse Arlt Institut o.A.); EasyNWK – zur computergestützten Erstellung von Netzwerkkarten (vgl. Pantucek 2012:189); EasyBiograph – zur computergestützten Erstellung von biografischen Zeitbalken (vgl. ebd.:228). Diese würden bereits Möglichkeiten und Potentiale aufzeigen, wie digitale Technologien in der Sozialen Arbeit genutzt werden können.

4.3.1. Interdisziplinarität

Vor allem wenn aus sozialwissenschaftlicher Forschung die Entwicklung von digitalen Tools hervorgehen soll, sei interdisziplinäre Zusammenarbeit für die technische Umsetzung notwendig. Hochschulintern gebe es daher Bemühungen zur studiengangsübergreifenden Zusammenarbeit. So werde auch am Ilse Arlt Institut interdisziplinäre Forschung bzw. Entwicklung angestrebt. Hier habe es bereits Kooperationen mit anderen Studiengängen, die die technische Umsetzung der drei oben genannten Tools ermöglichten, gegeben. Bei Brelomate habe sich gezeigt, wie sich Ansätze der Produktentwicklung, wie sie in der Programmierung angewandt werden, mit Methoden und Techniken der Sozialen Arbeit vereinen lassen. Mit Hilfe des Verfahrens des „User-Center-Designs“ wurde das Produkt an die Bedürfnisse der Nutzer*innen ausgerichtet. Das methodische Vorgehen in der Sozialen Arbeit, das Klient*innen als Expert*innen ihrer Lebenswelt erkennt, knüpft somit an die technische Nutzer*innenorientierung an.

4.3.2. Forschungsprojekte und -förderungen

Die Fachhochschule sei für die Forschung aus finanzieller Sicht auf Forschungsförderungen angewiesen. Diese würden diesbezüglich eine der wichtigsten Ressourcen darstellen. Zur Einreichung eines Forschungsprojektes brauche es eine eindeutige Leistungsbeschreibung und einen eindeutigen Zeitplan. Damit werde ein gewisser Rahmen vorgegeben, welche Tätigkeiten bis wann zu geschehen haben. Somit sei man auf die Abrufbarkeit gewisser Kompetenzen und Ressourcen zu bestimmten Zeitpunkten angewiesen. Förderungen würden durch die Finanzierung von Dienstverhältnissen diese Rahmenbedingungen ermöglichen.

Sollten Forschungsthemen, für die keine Förderung genehmigt wurde, dennoch als wichtig erachtet werden, können diese in Lehrforschungs-Projekte wie z.B. Bachelor- oder Masterarbeiten ausgelagert werden. Würden für ein Projekt auch Kompetenzen außerhalb der Sozialen Arbeit benötigt werden, so würden Kooperationen mit anderen Studiengängen genutzt werden, die ebenso Projekte auf Lehrforschungsbasis initiieren können. Diese Form der Ressourcenmobilisierung wurde im Interview mehrmals als Stärke beschrieben.

Entstehen aus der Forschung Produkte wie z.B. die vorhin beschriebenen digitalen Tools, dann ergeben sich aus der Finanzierungslogik gewisse Problematiken. Finanziert werde in erster Linie die Grundentwicklung eines Projektes an der Hochschule. Sollte aus einem Forschungsprojekt ein Produkt entstanden sein, kann dieses nicht weiter von der Hochschule betreut werden, da die finanziellen Aufwendungen dafür nicht gedeckt seien. Die Wartung des Produkts und das Anbieten von Serviceleistungen für Nutzer*innen müsse an eine andere Stelle – z.B. an ein Unternehmen – übergeben werden. Im Gegensatz zur Sozialen Arbeit würden in anderen Studiengängen aus Studierendenprojekten immer wieder Start-ups gegründet werden, die durch die Fachhochschule in Form des „Creative Pre-Incubator“ unterstützt werden würden. So könne zum einen das Projekt weiteren Nutzen finden, zum anderen könnten die Gründer*innen von ihrem Projekt profitieren. Dass auch die Weiterentwicklung und -betreuung der digitalen Tools des Ilse Arlt Institutes Nutzen hätten, zeige sich beispielsweise anhand von Anfragen von Organisationen, die Interesse an der Einbindung der digitalen Diagnostikinstrumente in ihr EDV-System zeigen. Für die weitere Betreuung solcher Produkte brauche es noch Lösungen. Wie eine solche mithilfe des Open-Source-Konzeptes aussehen könnte, wird im nächsten Kapitel beschrieben.

5. Resümee

An dieser Stelle sei die Forschungsfrage noch einmal dargelegt, um die Ergebnisse hinsichtlich der Fragestellung zusammenzufassen und zu diskutieren. Die Unterfragen dienen dazu, Schwerpunkte für die Hauptfrage zu setzen. Deshalb werden diese nicht speziell behandelt, sondern sie sind in der Beantwortung der Hauptfrage eingearbeitet. Die Forschungsfrage lautet:

„Welche Nutzungsmöglichkeiten für die Soziale Arbeit ergeben sich aus der Open-Source-Softwareentwicklung?“

Der Begriff Open Source, wie er eingangs definiert wurde, impliziert, dass die Software frei ist. Das bedeutet nicht nur, dass sie kostenfrei ist, sondern auch frei im Sinne der Nutzungsmöglichkeiten.

Open-Source-Software kann somit von den Nutzer*innen weiterentwickelt werden. Grundsätzlich können daher zwei Möglichkeiten unterschieden werden, wie sich Open-Source-Softwareentwicklung für die Soziale Arbeit nutzen lässt. Zum einen in der kostenfreien Verwendung der entstehenden Software und zum anderen in der Entwicklung von digitalen Tools. Diese beiden Zugänge werden in den nächsten beiden Unterkapiteln erörtert.

5.1. Nutzung von Open-Source-Software

Nutzer*innen digitaler Geräte und des Internets verwenden immer wieder und oftmals ohne, dass es ihnen bewusst ist, Open-Source-Software. Einerseits können Open-Source-Programme anwender*innen-seitig bewusst ausgeführte „Wirkapplikationen“ wie z.B. ein Mozilla-Firefox-Browser sein, andererseits auch im Hintergrund ablaufende „Supportapplikationen“ wie z.B. ein Apache-Webserver. Hinsichtlich der Forschungsfrage möchte ich das Augenmerk auf die Nutzung kostenfreier Wirkapplikationen legen.

Kostenfragen gehören für die Soziale Arbeit zum täglichen Geschäft, da zum einen Armut in vielen Handlungsfeldern der Gegenstand der Arbeit ist und zum anderen die Finanzierung von Organisationen zwangsläufig mit endlichen finanziellen Ressourcen verbunden ist. Die Kosten für Software können daher auf den Ebenen der Klient*innen und der Sozialarbeiter*innen bzw. der Organisationen Beachtung finden. Zur Veranschaulichung der Ebene der Klient*innen soll als Beispiel die Arbeit mit Menschen in finanziellen Problemlagen dienen. Die Lizenzgebühren für ein proprietäres Betriebssystem und ein Dokumentenbearbeitungsprogramm könnten bei der Neuanschaffung eines Computers die finanziellen Mittel von Klient*innen erschöpfen. Weiß die Beraterin oder der Berater um kostenfreie Open-Source-Alternativen wie das Ubuntu Betriebssystem und das Libre-Office-Dokumentenbearbeitungsprogramm Bescheid, könnten dahingehend finanzielle Belastungen vermieden werden. Parallel dazu können anhand der Nutzung von Open-Source-Software, als Alternative zu proprietärer Software, auf der Ebene sozialarbeiterischer Organisationen Kosten gespart werden.

Da die Anwendung von Open-Source-Programmen mit keinen Kosten verbunden ist, lädt sie zur innovativen Nutzung in der Zusammenarbeit mit Klient*innen ein. Beispielsweise könnte die Software „MediaWiki“ im Kontext sozialer Gruppenarbeit genutzt werden. MediaWiki ist eine Open-Source-Software, die als Grundlage für Wikipedia entwickelt wurde und kann als Plattform zur Informationssammlung, -verlinkung und zur kollaborativen Bearbeitung genutzt werden kann (vgl. MediaWiki 2015). Dabei könnten Betroffene eigene Erfahrungen aus Problemsituationen und

Lösungsansätze auf einer solchen Plattform weitergeben, um sich durch gemeinsames Wissen gegenseitig zu unterstützen.

Für die Tätigkeit im professionellen Kontext der Sozialarbeit bleibt bei der Verwendung digitaler Tools – unabhängig davon, ob es sich um Open-Source oder sonstige Software handelt – zu beachten, welche Daten wie und von wem verarbeitetet werden und ob sich dadurch datenschutzrechtliche Komplikationen ergeben. Diesbezüglich könnte zur Bewertung technischer oder auch juristischer Aspekte interdisziplinäre Zusammenarbeit notwendig sein. Liegt der Fokus auf der kostenfreien Nutzung, dann ist bei Open-Source-Software positiv zu betonen, dass sie meist in einem nicht-kommerziellen Kontext entstanden ist. Demgegenüber streichen Exner und Kratzwald bei der Nutzung von gratis Software(-Diensten) von kommerziellen Unternehmen hervor:

> „Während nämlich Unternehmen ein Interesse daran haben, Inhalte, also Musik, Filme, Informationen oder Codes unter Urheberrechtsschutz zu stellen, um damit Profit zu machen, haben sie ebenso großes Interesse daran, die Daten ihrer Nutzerinnen zu sammeln und darüber frei zu verfügen, um sie gewinnbringend zu verkaufen. Als Grundregel dabei gilt: Ist ein Dienst gratis, so ist die Nutzerin die Ware.“ (Exner / Kratzwald 2012:71)

Zusammengefasst bietet Open-Source-Software für die Soziale Arbeit die Möglichkeit der kostenfreien Nutzung von Programmen. Diese Ressource kann der Kostenersparnis hinsichtlich Lizenzgebühren für Organisationen wie auch für Klient*innen und weiters als innovatives Arbeitsmittel in der Zusammenarbeit mit Klient*innen dienen.

5.2. Open Source als Konzept zur Softwareentwicklung

Die am Ilse Arlt Institut entwickelten digitalen Tools Brelomate, EasyNWK und EasyBiograph zeigen erste Möglichkeiten auf, wie die Digitalisierung in der Sozialen Arbeit genutzt werden kann. Für die zukünftige Umsetzung von Software-Projekten soll auf die Möglichkeiten und Grenzen, die Open Source bietet, hingewiesen werden.

In der Sozialen Arbeit ist man für die technische Entwicklung solcher Tools auf interdisziplinäre Zusammenarbeit angewiesen, auf Personen mit den für die Softwareentwicklung benötigten technischen Kompetenzen. Vor allem wenn wenig finanzielle Mittel zur Verfügung stehen, wird die Frage laut, wie man diese Ressourcen mobilisieren kann. In Anlehnung an die in den Ergebnissen beschriebene Organisation und Arbeitsweise von Open Source lässt sich die Hypothese aufstellen, dass Open Source nicht dazu dient, über das Internet Programmierer*innen zu finden, die eine zuvor entwickelte Idee in Software umsetzten. Vielmehr muss bereits ein erstes Produkt

angeboten werden, um welches sich eine Community bilden kann, die in weiterer Folge an der Mitentwicklung partizipiert.

Am Ilse Arlt Institut wurde eine solche Grundentwicklung von Softwareprodukten im Zuge von Forschungsprojekten und in Zusammenarbeit mit technischen Studiengängen entwickelt. Nach der Grundentwicklung der Tools war eine weitere Produktbetreuung oder Weiterentwicklung nicht möglich, da diese Kosten durch die Forschungsförderung nicht gedeckt wurden. Hier soll die Möglichkeit aufgezeigt werden, ein Produkt aktiv zu halten, indem es als Open-Source-Projekt mit entsprechender Lizenzierung veröffentlicht wird. Damit soll angeregt werden, dass Nutzer*innen die Möglichkeit zur Mitentwicklung haben und das Produkt mitbetreuen können. Wobei hier nicht unmittelbar die Sozialarbeiter*innen angesprochen werden sollen, sondern deren Organisationen, die die EDV-Systeme zur Verfügung stellen. Das solidarökonomische Moment der Open-Source-Softwareentwicklung kann anhand der Frage aufgezeigt werden, welchen Nutzen ein Produkt bieten müsste, damit möglichst viele Personen an einer Mitarbeit Interesse hätten. Zur Anregung der Partizipation in der Softwareentwicklung wird daher nicht auf den Nutzen des Urhebers / der Urheberin, sondern auf den Nutzen aller verwiesen und somit ein reziprokes Verhältnis unter den Beteiligten geschaffen. Da im Bereich der Sozialen Arbeit digitale Tools wie auch das Open-Source-Konzept eine geringe Reichweite zu haben scheinen, müsste zur Errichtung einer Community auf mögliche Interessent*innen zugegangen werden. Am Beispiel der digitalisierten sozialen Diagnostikinstrumente zeigte sich bereits Interesse seitens Organisationen, die diese in ihr EDV-System einbinden wollten. Neben Organisationen könnten auch andere Hochschulen über eine Mitarbeit informiert werden. Um eine Beteiligung seitens Praktiker*innen und Organisationen zu erreichen, müsste der Nutzen eines Produktes klar formuliert und ersichtlich sein. Zeichnet sich für eine Organisation ein Nutzen ab, dann könnten finanzielle Mittel oder die bestehenden EDV-Abteilungen bzw. Programmierer*innen der Organisationen Ressourcen für das Projekt sein. Durch die Einbindung von im Feld der Sozialen Arbeit tätigen Personen und Organisationen gelangen zudem Praxiserfahrungen mit in das Projekt.

Sollte nach einer Grundentwicklung eines digitalen Tools kein Interesse vorhanden sein, dieses weiterhin zu betreuen, so könnte die Veröffentlichung des Sourcecodes dennoch dazu dienen, das Projekt als Basis für spätere, ähnliche Entwicklungen heranzuziehen. Im Kontext der Hochschule ergänzt sich das Konzept „Open Source“ passend mit den Bemühungen, das Wissen aus der Forschung als Common für einen gesellschaftlichen Nutzen zur Verfügung zu stellen.

Resümierend lässt sich festhalten, dass das Open-Source-Konzept eine Möglichkeit bietet, für die

Soziale Arbeit entwickelte Softwareprodukte kollaborativ zu betreuen und weiterzuentwickeln, indem man Nutzer*innen auf Ebene der Organisationen als Mitentwickler*innen einbindet und so vorhandene Ressourcen, wie beispielsweise angestellte EDV-Techniker*innen, nutzt. Eine Voraussetzung für ein solches Unterfangen wird sein, den Nutzen digitaler Tools ersichtlich zu machen und sich mit möglichen Interessent*innen zu vernetzen.

5.3. Reflexion der Ergebnisse und Ausblick

Für diese Bachelorarbeit gilt es festzuhalten, dass sie sich mit einer bis jetzt wenig bearbeiteten Thematik auseinandersetzt. Die auf Basis der Experteninterviews dargelegten Überlegungen zeigen Möglichkeiten auf, wie Open Source in der Sozialen Arbeit verwertet werden könnte. Die Ergebnisse legen keine genauen Konzepte oder Auflistungen von sozialarbeitsrelevanter Software dar. Vielmehr sind sie eine erste Orientierung in einem Verständnis von Sozialer Arbeit, mit der technische Aspekte und Möglichkeiten aufgezeigt und mitgedacht werden sollen. Gleichzeitig wird in Anbetracht begrenzter finanzieller Ressourcen nach innovativen Mitteln gesucht wie jene der solidarökonomischen Projekte. Für die Nutzung in der Sozialen Arbeit muss dabei kritisch hinterfragt werden, in welcher Position Solidarökonomie innerhalb des Sozialstaates verortet werden kann. Dienen sozial-innovative Projekte der Schwächung des Sozialstaates und als Nährboden für neoliberale Politik oder doch als notwendiger Rahmen, um sozialarbeiterische Konzepte wie Empowerment und Partizipation umsetzen zu können? Weinzierl und Novy merken an, dass vor dem Hintergrund neoliberaler Politik der Fokus von gesellschaftlichen Problemen auf eine Mikro-Ebene gelenkt werde. Die lokale Problemlösung durch sozial-innovative Projekte wie die der Solidarökonomie würde dazu führen, dass strukturelle Herausforderungen entpolitisiert und Ungleichverhältnisse aufrecht erhalten bleiben (vgl. Weinzierl / Novy 2016:126-127). „Kreativität und Engagement der Gesellschaft sollen genutzt werden, um Budgetengpässe auszugleichen“ (ebd.:127). Wagner bringt im folgenden Zitat einen weiteren Blickwinkel in die Diskussion:

> „Wirklich ‚innovativ‘ wäre demgegenüber, […] Lehren aus den Praxen der Leute für die Gestaltung einer sozialen Infrastruktur zu ziehen, die möglichst von allen bedingungslos zur Reproduktion des eigenen Lebens und zur Herzvorbringung des Sozialen genutzt werden kann.“ (Wagner 2016:207)

Das bedeutet, dass von politischer Ebene aus die Bottom-Up-Prozesse der Solidarökonomie bzw. der sozialen Innovation im Allgemeinen nicht als kostengünstige Problemlösungen, sondern als Hinweise auf Bedürfnisse nach Veränderungen oder Alternativen zu bestehenden Strukturen verstanden werden. Das widerspricht jedoch dem politischen Versuch, soziale Innovation top-down zu initiieren, wie er auch auf Ebene der EU verfolgt wird (vgl. Weinzierl / Novy 2016:126-127). Um

die Kluft zwischen politischer Verantwortungsabgabe und der Eigenermächtigung der Menschen zu schließen, könnte es einerseits hilfreich sein, für sozial-innovative Projekte staatliche Förderschienen zu schaffen, die in ihren Rahmenbedingungen offen genug sind, um das Bottom-Up-Verständnis aufrechtzuerhalten. Andererseits könnten, wie Wagner es fordert, die Initiativen als Hinweise für die Sozialpolitik wahrgenommen werden, um strukturelle Problemlagen politisch zu bearbeiten.

Um einen Ausblick auf weitere Forschung zu bieten, sei auf die Zweideutigkeit des in dieser Arbeit untersuchten Gegenstands hingewiesen. Mit den Nutzungsmöglichkeiten der solidarökonomischen Open-Source-Softwareentwicklung für die Soziale Arbeit wurde das Augenmerk vor allem auf die darin liegenden Chancen gelenkt. Die mit der Solidarökonomie wie auch mit der Digitalisierung verbundenen Risiken blieben dabei weitgehend unbehandelt und erfordern daher eine weitere Auseinandersetzung.

Literatur

Acksel, Britta / Euler, Johannes / Gauditz, Leslie / Helfrich, Silke / Kratzwald, Brigitte / Meretz, Stefan / Stein, Flavio / Tuschen, Stefan (2015): Commoning. Zur Kon-struktion einer konvivialen Gesellschaft. In: Adloff, Frank / Heins, Volker M. (Hrg.): Konvivialismus. Eine Debatte. Bielefeld: Transcript Verlag, 133-145.

Anastasiadis, Maria (o.A.): Solidarische Ökonomie. Bestandsaufnahme und Perspektiven in Österreich. http://studylibde.com/doc/2053545/solidarische-%C3%B6konomie [17.02.2018].

Anastasiadis, Maria / Essl, Günter / Riesenfelder, Andreas / Schmid, Tom / Wetzel, Petra (2003): Der Dritte Sektor in Wien. Zukunftsmarkt der Beschäftigung? http://www.sfs-research.at/projekte/P31-EQUAL%20Dritter%20Sektor/Der%20Dritte%20Sektor%20in%20Wien.pdf [17.02.2018].

Attac (o.A.): Solidarische Ökonomie. Selbstverständnis. http://www.attac-netzwerk.de/ag-solioeko/selbstverstaendnis/ [17.02.2018].

Embshoff, Dagmar / Giegold, Sven (2008): Solidarische Ökonomie im globalisierten Kapitalismus. In: Embshoff, Dagmar / Giegold, Sven (Hrg.): Solidarische Ökonomie im globalisierten Kapitalismus. Hamburg: VSA-Verlag, 11-24. http://www.vsa-verlag.de/uploads/media/VSA_Giegold_ua_Solidarische_Oekonomie_komplett.pdf [16.02.2018].

Exner, Andreas / Kratzwald, Brigitte (2012): Solidarische Ökonomie & Commons. INTRO. Wien: Mandelbaum kritik & utopie.

Fricke, Klaus (2013): Digitaltechnik. Lehr- und Übungsbuch für Elektrotechniker und Informatiker. 7. Auflage, Wiesbaden: Springer Fachmedien.

Grassmuck, Volker (2004): Freie Software. Zwischen Privat- und Gemeineigentum. 2. Auflage, Bonn: Bundeszentrale für politische Bildung. http://freie-software.bpb.de/Grassmuck.pdf [20.02.2018].

Gumm, Heinz Peter / Sommer, Manfred (2011): Einführung in die Informatik. 9. Auflage, München: Oldenbourg Verlag.

Helfrich, Silke (2013): Commoning als Strategie der Armutsvermeidung. In: Die Armutskonferenz (Hrg.): Was allen gehört. Commons – Neue Perspektiven in der Armutsbekämpfung. Wien: Verlag des Österreichischen Gewerkschaftsbundes GesmbH, 27-44. http://www.albanknecht.de/publikationen/buch_was-allen-gehoert_web.pdf [18.03.2018].

Ilse Arlt Institut (o.A.): Forschung. Brelomate - Mehr Abwechslung und Kommunikationsmöglichkeiten für ältere Menschen mit technologischer Unterstützung. http://inclusion.fhstp.ac.at/index.php/forschung/projekte/299-brelomate [21.03.2018].

Kharitoniouk, Svetlana / Stewin, Patrick (2004): Einleitung. In: Gehring, Robert / Lutterbeck, Bernd (Hrg.): Open Source Jahrbuch 2004. Zwischen Softwareentwicklung und Gesellschaftsmodell. Berlin: Lehmanns Media, 1-16. http://www.opensourcejahrbuch.de/download/jb2004/OpenSourceJahrbuch2004.pdf [20.02.2018].

Krotz, Friedrich (2001): Die Mediatisierung kommunikativen Handelns. Der Wandel von Alltag und sozialen Beziehungen, Kultur und Gesellschaft durch die Medien. Wiesbaden: Westdeutscher Verlag.

Mayring, Philipp (2015): Qualitative Inhaltsanalyse. Grundlagen und Techniken. 12. Auflage, Weinheim und Basel: Beltz Verlag.

MediaWiki (2015): Willkommen bei MediaWiki.org. https://www.mediawiki.org/wiki/MediaWiki/de [03.03.2018].

OSI - Open Source Initiative (o.A.): Licences. Open Source Licenses by Category. https://opensource.org/licenses/category [17.03.2018].

OSI - Open Source Initiative (2007): Licences. The Open Source Definition. https://opensource.org/osd [21.02.2018].

Pantucek, Peter (2012): Soziale Diagnostik. Verfahren für die Praxis Sozialer Arbeit. 3. Auflage, Wien-Köln-Weimar: Böhlau Verlag.

Precht, Manfred / Meier, Nikolaus / Tremel, Dieter (2004): EDV-Grundwissen. Eine Einführung in Theorie und Praxis der modernen EDV. 7. Auflage, München: Addison-Wesley Verlag.

Raymond, Eric (1999): Die Kathedrale und der Basar. http://www.selflinux.org/selflinux/pdf/die_kathedrale_und_der_basar.pdf [20.02.2018].

Rösch, Eike (2018): OpenSource-Tools als pädagogische Chance. In: Explizit, Fachmagazin offene Jugendarbeit, 01/2018, 27-28. http://www.boja.at/fileadmin/download/Service/bOJA_Explizit_2018.pdf [21.02.2018].

Smid, Volker (2006): Novell goes Open Enterprise. In: Lutterbeck, Bernd / Bärwolff, Matthias / Gehring, Robert (Hrg.): Open Source Jahrbuch 2006. Zwischen Softwareentwicklung und Gesellschaftsmodell. Berlin: Lehmanns Media, 19-30. http://www.opensourcejahrbuch.de/download/jb2006/OpenSourceJahrbuch2006_online.pdf [24.02.2918].

Stallman, Richard (2007): Warum „Open Source“ das Wesentliche von „Freier Software“ verdeckt. Von der Freie-Software-Bewegung. In: Lutterbeck, Bernd / Bärwolff, Matthias / Gehring, Robert (Hrg.): Open Source Jahrbuch 2007. Zwischen freier Software und Gesellschaftsmodell. Berlin: Lehmanns Media, 1-7. http://www.opensourcejahrbuch.de/download/jb2007/OpenSourceJahrbuch2007_online.pdf [20.02.2018].

Steiner, Oliver (2017): Mediatisierung und Soziale-Arbeit – what’s next? Der Einsatz digitaler Technologien ist von grundsätzlichen Ambivalenzen geprägt. In: SozialAktuell, 05/2017, 8-12.

Talenteverbund (o.A.): Was ist Cyclos. https://talenteverbund.at/item/108-was-ist-cyclos [30.01.2018].

Wagner, Thomas (2016): Aus der Not heraus? Armut, soziale Ausschließung und Wohlfahrtsproduktion „von unten“. In: Meichenitsch, Katharina / Neumayr, Michaela / Schenk, Martin (Hrg.): Neu! Besser! Billiger! Soziale Innovation als leeres Versprechen? Wien: Mandelbaum Verlag, 198-210.

Weinzierl, Carla / Novy, Andreas (2016): Partizipation und Empowerment in sozialen Innovationen. In: Meichenitsch, Katharina / Neumayr, Michaela / Schenk, Martin (Hrg.): Neu! Besser! Billiger! Soziale Innovation als leeres Versprechen? Wien: Mandelbaum Verlag, 125-136.

Widmer, Ursula / Bähler, Konrad (2006): Open-Source-Lizenzen - Wesentliche Punkte für Nutzer, Entwickler und Vertreiber. In: Lutterbeck, Bernd / Bärwolff, Matthias / Gehring, Robert (Hrg.): Open Source Jahrbuch 2006. Zwischen Softwareentwicklung und Gesellschaftsmodell. Berlin: Lehmanns Media, 165-179. http://www.opensourcejahrbuch.de/download/jb2006/OpenSourceJahrbuch2006_online.pdf [24.02.2918].

Wikipedia (2018): Wikipedia. https://de.wikipedia.org/wiki/Wikipedia [30.01.2018].

Wöstenkühler, Gerd (2016): Grundlagen der Digitaltechnik. Elementare Komponenten, Funktionen und Steuerungen. 2. Auflage, München: Carl Hanser Verlag.

Daten

I1, narratives Interview, geführt von Clemens Rosenthaler mit einem Informatik Dozenten, 17.11.2017, Dauer 59:10, Audiodatei.

I2, leitfadengestütztes Interview, geführte von Clemens Rosenthaler mit FH-Prof. Mag. Dr. Johannes Pflegerl, 29.01.2018, Dauer 37:28, Audiodatei.

I3, leitfadengestütztes Interview, geführte von Clemens Rosenthaler mit einem IT-Unternehmer, 01.02.2018, Dauer 1:02:35, Audiodatei.

T1, Transkript von Interview I1, erstellt von Clemens Rosenthaler, 29.11.2017, Zeilen nummeriert.

T2, Transkript von Interview I2, erstellt von Clemens Rosenthaler, 02.02.2017, Zeilen nummeriert.

T3, Transkript von Interview I3, erstellt von Clemens Rosenthaler, 03.02.2018, Zeilen nummeriert.

Eidesstattliche Erklärung

Ich, Clemens Rosenthaler, geboren am 18.06.1995 in St. Pölten, erkläre,

1. dass ich diese Bachelorarbeit selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und mich auch sonst keiner unerlaubten Hilfen bedient habe,

2. dass ich meine Bachelorarbeit bisher weder im In- noch im Ausland in irgendeiner Form als Prüfungsarbeit vorgelegt habe,

St. Pölten, am 26.03.2018