universität wien

MASTERARBEIT

Titel der Masterarbeit

„Die Organisation gemeinsamer Wissensproduktion im Internet“

Fallanalyse eines Projekts der freien/offenen Softwareentwicklung.

Verfasser

Laurens Georg Lauer, B.A.

angestrebter akademischer Grad

Master of Arts (MA)

Wien, 2012

Studienkennzahl lt. Studienblatt:	A 066 905
Studienrichtung lt. Studienblatt:	Soziologie
Betreuerin / Betreuer:	Ao. Univ.-Prof. Mag. Dr. Ulrike Froschauer

Inhaltsverzeichnis

1 Einleitung

Gegenstand der vorliegenden Arbeit ist die Fallanalyse eines Projekts der freien/offenen Softwareentwicklung im Internet. Das noch relativ junge Phänomen der freien/offenen Softwareentwicklung ist eine zumeist ausschließlich über das Verbreitungsmedium Internet vermittelte Form der sozialen Kooperation, welche die Beteiligung und Zusammenarbeit einer Vielzahl von AkteurenInnen[1] zur gemeinsamen Produktion von Software organisiert und koordiniert. Die freie/offene Softwareentwicklung markiert dabei ein gesellschaftliches Feld, das sich in Abgrenzung zur proprietären Softwareindustrie über grundlegende Wertvorstellungen der rechtlichen, ökonomischen, ethischen und technischen Aspekte von Software und deren Entwicklung konstituiert und im Rahmen eines öffentlichen Diskurs thematisiert. Insbesondere der Erfolg einiger prominenter Softwareprogramme (z.B. das Betriebssystem Linux, der Apache-HTTP-Server oder der Mozilla Browser) haben das öffentliche Interesse angeregt und zu einer erheblichen Zunahme offener/freier Softwareprojekte geführt. Diese organisieren sich unter denkbar unwahrscheinlichen Bedingungen: Die Beteiligten sind global verteilte Akteure, die ihren Kontakt häufig ausschließlich über schriftliche Kommunikation im Internet realisieren. Der Zugang zum Arbeitsgegenstand Software ist aufgrund von Wertvorstellungen sowie rechtlichen Lizenzbedingungen für alle Personen gesichert, jedoch nur sehr bedingt in ökonomische Verwertungszusammenhänge zu überführen. Dies setzt eine Beteiligung auf unentgeltlicher Basis voraus, die durch eine freiwillige, der Zusammensetzung der Individuen entsprechend heterogene, Motivgrundlage charakterisiert ist. Nichtsdestotrotz etablieren sich im Internet vielfältige Projekte, die über Jahre hinaus die Entwicklung einer Software vorantreiben und stetig neue Versionen kostenlos veröffentlichen, nicht selten mit einem aussergewöhnlich hohen Qualitätsstandard. Dabei zeigt die „Community“ – sowohl ideologisch (in der Außendarstellung) als auch praktisch – eine hohe Offenheit für Außenstehende, sei es für Verbesserungsvorschläge, Fehlermeldungen oder Hilfsgesuche von Laien im Umgang mit der Software oder Quellcodebeiträge und –veränderungen „fremder“ Entwickler. Über die entsprechenden Webportale kann sich potenziell jeder an die Entwickler der Software wenden und findet in der Regel innerhalb kürzester Zeit auch Anschluss. Neben der nicht unerheblichen Komplexität der Software selbst, die es in einer gemeinsamen Entwicklung zu bearbeiten gilt, treten auf diese Weise noch zusätzliche Anforderungen und Aufgaben an die Projekte heran und erweitern den Koordinationsbedarf bzw. die –leistung erheblich. Wie organisieren Projekte der freien/offenen Softwareentwicklung unter diesen

[1] Im Folgenden wird statt der orthografischen Binnen-I Praxis stets die männliche Form benutzt. Dieses Vorgehen soll die Bedeutung weiblicher Akteure im Rahmen freier/offener Softwareentwicklung nicht marginalisierten und dient auch nicht ausschließlich der besseren Lesbarkeit, sondern sollte vielmehr als Verweis auf die mit 98 Prozent sehr deutliche Überrepräsentation männlicher Entwickler verstanden werden (Gosh et al. 2002).

Bedingungen ihre sozialen Zusammenhänge? Was für Konfigurationen der Kooperation lassen sich identifizieren und wie lassen sich diese als soziale Ordnungsleistung erfassen und verstehen?

Vor diesem Hintergrund bietet die Systemtheorie nach Niklas Luhmann einen analytisch-heuristischen Bezugsrahmen für die Untersuchung der sozialen Projektorganisation freier/offener Softwareentwicklung: Ausgehend von Kommunikation unter den Bedingungen doppelter Kontingenz lassen sich soziale Systeme als Selektionszusammenhänge besonderer Art konzipieren, die sich durch ihre operationalen Prozesse von der Umwelt abgrenzen und durch (Erwartungs-)Strukturen stabilisieren (Luhmann 1981a). Die Konstitution und (Re-)Produktion des Systems findet damit ihren Bezugspunkt in den Formen emergenter Sinnordnungen, die als abgestimmte Selektion von Strukturen und Prozessen die Umweltkomplexität durch den Aufbau systemeigener Komplexität reduziert und organisiert (Luhmann 1990a, 1987).

Methodisch, d. h. aus einer funktionalen Perspektive, rücken damit die (System-) Leistungen in den Fokus, welche die Organisation und Stabilisierung der System/Umwelt-Relation im Sinne relativer Invarianz der Systemordnung und der Systemgrenzen gegenüber einer komplexeren Umwelt realisieren. Die strukturierte Einheit des Systems lässt sich demnach nur in Bezug auf die Einheit der Differenz System/Umwelt betrachten: Spezifische Formausprägungen des Systems können unter diesem Gesichtspunkt dann auf ihre Ordnungsleistung hin erfasst und analysiert sowie auf ihre Funktion und Folgeprobleme hin interpretiert werden (Luhmann 1974a). Eine solche Methodik setzt nicht in Form einer empirischen Hypothese an, sondern zielt auf eine möglichst umfassende Untersuchung der Systemstrukturen als emergente Sinnordnung. Dieses Vorgehen zielt letztlich auf einen übergreifenden Vergleich funktionaler Äquivalenzen, der in einer Einzelfallanalyse nur beschränkt zu realisieren ist: Die Betrachtungen bleiben hier stärker auf den Problemkontext des untersuchten Systems bezogen und müssen sich entsprechend der Abstraktion ihres Ansatzes bewusst sein (Luhmann 1974b). Dafür ermöglichen sie eine detaillierte Betrachtung der realisierten Zusammenhänge, die sich in Rückgriff auf den elaborierten theoretischen Rahmen der Systemtheorie abstrahieren und einer Interpretation zuführen lassen.

Methodologisch fußt der Ansatz auf einer erkenntnistheoretischen Position des Konstruktivismus. Das heißt: Alle Beobachtungen, auch die des Forschers, unterliegen den Bedingungen der autopoietischen, selbstreferenziellen Schließung des erkennenden Systems und alle Realität wird kognitiv über Unterscheidungen konstruiert (Luhmann 2001, 1990b). Als (Selbst)Beobachtungen zweiter Ordnung ermöglichen sie das laufende Beobachten von Beobachtungen und damit der genutzten Unterscheidung(en) selbst. Soziologisch richtet sich der Fokus dann „auf die für den

beobachteten Beobachter latenten Strukturen und Funktionen“ (Luhmann 1990b: 46), an die es konsistent in der Betrachtung der sozialen Zusammenhänge anzuschließen gilt, und zwar hier aus einer verstehenden Perspektive, d.h. unter der Leitdifferenz der System/Umwelt-Differenz des beobachteten Systems (Luhmann 1986a).

Vor diesem Hintergrund zielt die Untersuchung des Projekts der freien/offenen Softwareentwicklung auf die organisationalen Strukturen des Kooperationszusammenhanges, die es im Sinne eines System/Umwelt-Gefüges zu analysieren und zu verstehen gilt. Dazu werden im zweiten Kapitel in Rückgriff auf die bestehende Forschung die konstitutiven Sinnprämissen der freien/offenen Softwareentwicklung als soziale Bewegung rekonstruiert und zusammenfassend herausgearbeitet. Sie stellen als solche den kontextualen Rahmen der Projektorganisation dar, der in diesen eine operative Form findet und sich selbst (re-)produziert. An diese Grundlegung schließt sich die exemplarische Untersuchung eines Projekts der freien/offenen Softwareentwicklung im Internet an: Im dritten Kapitel gilt es zunächst, die spezifischen Mediumsqualitäten der internet-basierten Kommunikationstools der Projektplattform konzeptionell zu erfassen, um diese als Gebrauchsgegenstände im Rahmen einer (exemplarischen) Artefaktanalyse zur (Re-)Konstruktion kollektiver bzw. emergenter Sinnstrukturen des organisationalen Zusammenhanges heranzuziehen (Froschauer 2009; Lueger 2010). Die Tools stellen die zentralen Medien bzw. Träger der Projektkommunikation dar und ermöglichen anhand der Betrachtung ihres Aufbaues grundlegende Einsichten in die Regulation der Projektorganisation und die dahinter stehende „Logik“. Anschließend richtet sich der Fokus auf das relationale Gefüge der Projektkommunikation, die sich in Form eines Netzwerkes erfassen und auf strukturelle Ausprägungen im Sinne sozialer Figurationskonstellationen hin untersuchen lassen. Dazu wird im vierten Kapitel zunächst das Konzept sozialer Netzwerke dargelegt und in Beziehung zum systemtheoretischen Verständnis sozialer Systeme gesetzt, um eine gemeinsame Basis theoretischer Anknüpfpunkte für die Analyse organisationaler Formen und Konfigurationen sowie deren Interpretation zu entwickeln. Auch methodologisch lassen sich Unterschiede – zwischen qualitativ rekonstruierender und quantitativ messender Herangehensweise – ausmachen, die jedoch als sich ergänzende Perspektiven behandelt und nicht ausdrücklich gegenübergestellt werden. Der hier verfolgte Ansatz beruht auf der Annahme, dass die empirisch erfassten Beziehungsstrukturen sich als beobachtbare soziale Manifestation in Beziehung zu einem bedeutungsgenerierenden sozialen Kontexts verstehend deuten lassen (und im Sinne einer konsistenten Argumentation auch als Kontrollinstanz dienen können). Entsprechend folgt im fünften Kapitel zunächst die Dokumentation der Datenerhebung und der empirischen Inspektion des Kommunikationsnetzwerkes, bevor die Ergebnisse in einem umfassenden Fazit vor dem Hintergrund der vorausgehenden

Ausführungen verortet und interpretiert werden. Die Arbeit schließt mit einem kurzen Resümee, das grundlegende Leerstellen und Anschlussmöglichkeiten für weitere Forschung aufzeigt.

2 Die freie/offene Softwareentwicklung

2.1 Rahmung

Soziale Systeme[2] realisieren sich unter kontextualen Rahmenbedingungen, welche ihre Form und Struktur prägen. Ermöglichungen und Begrenzungen sozialer Ordnungsformen im Internet entwickeln sich aus bzw. in Umwelten mit technologischen, rechtlichen, ökonomischen und kulturellen Strukturen, die im Bereich der freien/offenen Softwareentwicklung ein relativ neues und eigenständiges Gefüge darstellen. Wenn damit nicht „Einzelfälle und situative Lagen, sondern gesellschaftlich wiederkehrende Bedingungen angesprochen sind, ist von generalisierten Strukturen die Rede, für die sich zeigen lässt, dass sie selektiv auf organisatorische Formbildungen und ihre Reproduktionsmöglichkeiten wirken“ (Apelt/Tacke 2012: 14). Eine soziologische Analyse der Kooperationsformen und organisationalen Strukturen von freien/offenen Softwareprojekten im Internet muss in diesem Sinne die kontextualen Strukturen, in die diese Projekte eingebettet sind, aufzeigen. Dabei rücken zentral die kulturellen bzw. ideologischen Diskurse sowie rechtliche Regelungen und Bestimmungen in den Fokus. Verbunden sind damit mediale Aneignungsprozesse, Rollenvorstellungen und normative Erwartungen, die auf geteilte Sinn- und Relevanzstrukturen verweisen. Soziale Ordnungsleistungen der Projekte freier/offener Softwareentwicklung entfalten sich in diesem Rahmen und wirken auf ihn zurück.

Im Folgenden werden anhand bestehender Forschungsergebnisse die wichtigsten Rahmenbedingungen sozialer Systembildung im Bereich der freien/offenen Softwareentwicklung im Internet dargelegt. Die Rezeption der entsprechenden Untersuchungen, die sowohl Einzelanalysen als auch ausführliche Reviews umfassen, folgt einer theorieoffenen Ausrichtung, um sich nicht durch paradigmatische Festlegungen einer verkürzenden Perspektive auszusetzen. Ihre Ergebnisse werden in thematisch eigenständigen Abschnitten möglichst knapp zusammengefasst, um sie dann im Anschluss in einem systemtheoretischen Rahmen zu verorten und zu interpretieren.

Der Bereich der freien/offenen Softwareentwicklung lässt sich aus soziologischer Perspektive konzeptionell als soziale Bewegung erfassen und beschreiben (Zimmermann 2004; Holtgrewe 2000; Elliot/Scacchi 2008; Carillo/Akoli 2008). Die bestehende Forschung zeigt, dass sich die freie/offene Softwareentwicklung zu einem sozialen Phänomen entwickelt hat, welches elaborierte kollektive Identitäten mit geteilten Wertvorstellungen und Selbstbeschreibungen aufweist (Lehmann 2004). In Abgrenzung zur proprietären Softwareproduktion etabliert sich ab Anfang der 1980er

[2] Der Begriff „soziales System“ wird hier als umfassende Bezeichnung für verschiedene soziale Aggregats- bzw. Organisationsformen verwendet und dient noch nicht als klassifizierende Bestimmung. Diese Frage ist Teil des Forschungsanliegens und wird im weiteren Verlauf der Arbeit wiederkehrend behandelt.

Jahre ein zunehmend breiter Diskurs mit unterschiedlichen Semantiklinien und starker Prägung durch prominente Einzelakteure, der eine eigenständige Arbeitspraxis mit allgemeinen Regeln und Normen sowie mit unterschiedlichen argumentativen Begründungen bzw. Legitimationen hervorbringt.

Anhand zentraler Selbstthematisierungen der freien/offenen Softwareentwicklung lassen sich in Rückgriff auf bestehende Forschungen die zentralen Relevanz- und Sinnstrukturen dieses diskursiv konstruierten Gegenstandes herausarbeiten und zusammenfassen. In diesem Prozess geraten dabei „unweigerlich, weil konstitutiv für Identitäten, die Grenzziehungen nach außen und auch innerhalb einzelner Fraktionierungen in den analytischen Blick“ (Sebald 2008: 60). Die Ergebnisse werden nach zentralen Diskurslinien differenziert und auf ihre Ausprägung im Hinblick auf übergreifende[3], elementare Sinnprämissen hin untersucht, die den kontextualen Rahmen freier/offener Softwareprojekte bilden.

2.2 Geschichtlicher Hintergrund

Das Phänomen der freien/offenen Softwareproduktion ist eng mit der technischen Entwicklung des Computers und des Internets verbunden, die sich überwiegend an universitären und militärischen Einrichtungen vollzog. Die beteiligten Wissenschaftler etablierten eine ausgeprägte Kooperationspraxis, in der technische Entwicklungen und Softwareerzeugnisse untereinander ausgetauscht sowie von verschiedenen Akteuren weiterentwickelt und für ihre Zwecke verändert wurden (von Krogh/von Hippel 2003; Himanen 2001; Freyermuth 2007). Diese Praxis („Hackerkultur“) wurde erstmals im Jahre 1969 von IBM durch die Einführung von Softwarelizenzen zur Schaffung ökonomischer Verwertungsmöglichkeiten in Frage gestellt – eine Entwicklung, die in den nächsten 20 Jahren durch die Mehrzahl der existierenden Unternehmen in der Computer-Branche aufgegriffen und weitergetragen wurde (Freyermuth 2007). Software wurde nun als urheberrechtliches Eigentum behandelt, deren (zeitlich begrenzter) Gebrauch für den Abnehmer mittels Lizenzen kostenpflichtig reglementiert wurde und ausschließlich die (zeitlich begrenzte) Nutzung, nicht aber die Veränderung und Weitergabe der Software erlaubte. Der Quellcode der Software, der das essentielle Medium des Programmierens darstellt, wurde zurückgehalten und als Geschäftsgeheimnis behandelt (Freyermuth 2007; Lemley 1995). Die ursprüngliche „Hackerpraxis“ der Softwareentwicklung wurde so zunehmend unterbunden und in kapitalistische Produktionsverhältnisse überführt.

Im Jahre 1983 veröffentlichte das Unternehmen AT&T die Version „System V“ seines bis dahin kostenlosen und offenen Betriebssystems UNIX, welches zu diesem Zeit-

[3] Der Diskurs der freien/offenen Softwareentwicklung ist durch grundlegende semantische Differenzen geprägt, verfügt jedoch in der Entwicklerpraxis über recht einheitliche Sinnstrukturen, die als solche als einheitliches Gefüge behandelt werden.

punkt eine weitverbreitete Arbeitsgrundlage in Forschungseinrichtungen an verschiedenen Universitäten war, als proprietäre Version. In Reaktion auf dieses Ereignis gründete der am Massachusetts Institute of Technology (MIT) angestellte Programmierer Richard Stallman das Projekt „GNU“ (*G*NU is *N*ot *U*nix) mit dem Ziel, ein alternatives Betriebssystem zu entwickeln (Williams 2002). Zusammen mit der in den folgenden Jahren geschaffenen GNU General Public License und der Veröffentlichung eines GNU-Manifestes markiert dieses Ereignis den Beginn der freien Softwarebewegung.

Stallmans Softwareprojekt wies in den folgenden Jahren eine große Produktivität auf, die Entwicklung des Betriebssystem-Kerns verlief jedoch schleppend. Im Jahre 1991 bemerkte der Informatikstudent Linus Torvald, dass eine von ihm zu Studienzwecken programmierte Software sich als Grundlage für einen solchen Betriebssystem-Kern eignen würde, und rief im Internet zur freien Mitarbeit auf. Als Torvald seine Software im Jahre 1992 unter Stallmans GNU License stellte, wurde sie in die bisherigen Arbeiten der Stallman-Gemeinschaft auch rechtlich integrierbar, und im Jahre 1994 konnte die erste stabile Version des freien Betriebssystems Linux veröffentlicht werden (Moon/Sproull 2000).

Die Entwicklungsgeschichte von Linux wurde begleitet von einer intensiven Auseinandersetzung im Internet über das praktizierte Kooperationsmodell, deren prominentestes Ereignis die öffentlich im Internet geführte Diskussion zwischen Torvald und Andrew Stuart Tannenbaum, Professor für Informatik an der Freien Universität Amsterdam, ist (DiBona et al. 1999). Die in diesem Rahmen angestellten Überlegungen von Torvald prägten das Arbeitsverständnis freier/offener Softwareentwickler maßgeblich und wurden insbesondere von Erik S. Raymond, einem aktiven Entwickler und Publizist, aufgegriffen. Sein Essay „The Cathedral and the Bazaar“ (Raymond 2001) fand große Verbreitung und stellt eine Art übergreifendes Manifest der offenen Softwaregemeinde dar. Gemeinsam mit zwei weiteren Akteuren gründete Raymond im Jahre 1998 die Open Source Initiative, welche in Abgrenzung zur *freien* Software verstärkt die Effizienzvorteile des offenen Entwicklungsprozesses hervorhob und die wertgeladene, gesellschaftlich orientierte Argumentation der freien Softwareentwicklung ablehnte. Beide Bereiche weisen zwar nur wenige Unterschiede in ihren Definitionen freier bzw. offener Software, den Kooperationsvorstellungen und rechtlichen Lizenzen auf, sie unterscheiden sich jedoch maßgeblich in ihren Legitimationsgrundlagen und repräsentieren in diesem Sinne auch unterschiedliche ideologische Vorstellungen.

2.3 FLOSS-Diskurs

Die freie/offene Softwareentwicklung baut, insbesondere in ihren Anfängen, auf einem grundlegend positiven Technikverständnis auf. Mit computerbasierten Techno-

logien wurde der Glaube an weitreichende Fortschrittsmöglichkeiten verbunden, die sich unter den damaligen Pionieren gerade auch mit sozialpolitischen Hoffnungen verbanden. „Computerisierung“ galt als ‚prinzipiell für niemanden nachteilig‘ und konzeptionell offen für jeden Anwendungsbereich (Kling/Iacono 1988). Vor diesem Hintergrund markierte das GNU–Manifest von Richard Stallman den Ausgangspunkt des Diskurses der freien/offenen Software (Stallman 2012a). Die zentrale Thematik dieser Deklaration ist, wie in der Namensgebung „Free Software“ angelegt, die Freiheit, die ideengeschichtlich auf Vorstellungen individueller Grundrechte und des Allgemeingutes bezogen wird. Stallmans Argumentation setzt an einem Verständnis der wissenschaftlichen Erkenntnis als Allgemeingut an, wie es in der Grundlagenforschung üblich ist, und richtet sich gegen Verfügungsgewalten und Einschränkungen im Umgang mit dieser Erkenntnis. Dies bezieht sich jedoch nicht primär auf ökonomische Verwertungszusammenhänge und -zwänge, sondern auf die Ungebundenheit der Information an sich, d. h. die prinzipielle Möglichkeit und das Recht auf Zugang zu und selbstbestimmten Umgang mit vorhandenen Wissensvorräten (Bretthauer 2001; Carillo/Okoli 2008). Für Stallman ist freie Software „a matter of liberty, not price. To understand the concept, you should think of ‘free’ as in ‘free speech,’ not as in ‘free beer’“ (Stallman 2012b). Die Notwendigkeit der Informationsfreiheit wird dabei zentral über den Schutz und die Rechte des Individuums als eines selbstbestimmten und ungebundenen Wesens legitimiert. Die Freiheit des Softwareentwicklers – so die zentrale Überzeugung – lässt sich nur über einen frei zugänglichen Wissensvorrat realisieren bzw. sichern, der auch für den sozialen Ertrag des technischen Potenzials bestimmend ist (Sebald 2008; Stewart/Gosian 2006; Carillo/Okoli 2008).

Für den Bereich der (freien) Softwareentwicklung wird dabei weniger der Produktionsprozess als solcher thematisiert, sondern der Distributionsprozess und seine rechtlichen Aspekte werden als relevanter Bezugspunkt artikuliert (Sebald 2008). Als primäres Problem werden die Geheimhaltung des Quellcodes und Nutzungseinschränkungen durch proprietäre Softwarelizensierung, hier insbesondere das Verbot der Weitergabe, spezifiziert. Die definierten Bedingungen freier Software – das Recht auf das Studium, die Veränderung und Verbreitung von Software und Quellcode (Stallman 2012b) – setzten entsprechend den unbeschränkten Zugang zum Quellcode voraus und sichern diesen auch auf rechtlicher Basis[4] (Bretthauer 2001; Jaeger/Metzger 2002). Kooperationsaspekte werden in diesem Rahmen fast ausschließlich unter dem Aspekt des Teilens von Informationen thematisiert und entsprechend als kausal-logische und moralische Notwendigkeit konstruiert: Ausgangslage ist der selbstbestimmte, vornehmlich autonom arbeitende Softwareentwickler, der seine Ar-

[4] Siehe Kap. 2.5.

beit und seine Erkenntnisse freiwillig weitergibt und teilt (Carillo/Okoli 2008; Sebald 2008; Stewart/Gosain 2006).

Ab den späten 1980er Jahren entwickelte sich parallel zum Diskurs der freien Software unter den Softwareentwicklern der zunehmend vernetzten Rechenzentren der Universitäten und sonstigen Forschungseinrichtungen eine zweite Semantiklinie, die zentral die Praxis eines kooperativen Programmierens thematisierte (Sebald 2008). Prägendes Ereignis dieses Diskurses war die Entwicklung des Betriebssystems Linux um Linus Torvald, welches entscheidend für die Identitätsbildung und Selbstständigkeit dieser diskursiven Praxis ist, in dessen Rahmen sich jedoch auch ein grundlegender Anschluss an die freie Softwareentwicklung vollzieht. Die in diesem Rahmen sich konstituierende Argumentation entbehrt fast vollständig gesellschaftlicher bzw. moralischer Bezüge und fokussiert primär das Produktionsmodell der Softwareentwicklung (Bretthauer 2001). Technik gilt als grundsätzlich a-soziale Entität, die durch eine neutrale Eigenlogik bestimmbar und erfassbar ist. Als Wertideal dient in diesem Sinne die technische Qualität der Software, ihr kommt herausragende Bedeutung zu (Carillo/Okoli 2008; Feller/Fitzgerald 2000; Lehmann 2004; Stewart/Gosain 2006). Während die freie Softwareentwicklung noch tendenziell die individuelle Produktion abgeschlossener Themenbereiche bzw. Softwarekomponenten präferiert, praktiziert und legitimiert die Linux-Gemeinschaft eine grundsätzliche Offenheit auch im Kooperationsprozess der Softwareentwicklung (Sebald 2008). Als primäre Leitorientierung wird das Kriterium der „objektiv“ besten technischen Lösung konstruiert, vor dessen Hintergrund sich ein Kooperationsverständnis als Codeproduktion und -zusammensetzung nach ausschließlich technischen Kriterien ableitet. Torvald verzichtet – zumindest ideologisch – auf jede Kontrolle über die Programmierziele und -tätigkeiten der einzelnen Akteure und postuliert in der öffentlichen Auseinandersetzung mit dem arrivierten Informatikprofessor Tannenbaum eine für das damalige Verständnis völlig unorthodoxe Kooperationsform:

> Tannenbaum: „I think co-ordinating 1000 prima donnas living all over the world will be as easy as herding cats […]. If Linus wants to keep control of the official version, and a group of eager beavers want to go off in a different direction, the same problem arises […]. Anyone who says you can have a lot of widely dispersed people hack away on a complicated piece of code and avoid total anarchy has never managed a software project.”
>
> Torvald: „Just so that nobody takes his guess for the full truth, here’s my standing on ‘keeping control’, in 2 words (three?): I won’t.” (Tannenbaum/Torvald 1992)

Die hier verfolgte Argumentation setzt an einer Vorstellung technischer Objektivität an und konstituiert die Qualität der Software als fundamentale Leitorientierung, der entsprechend dem technischen Wissen und der praktischen Arbeit die höchste (personale) Wertigkeit zugeschrieben wird (Carillo/Okoli 2008; Ljunberg 2000; Sebald

2008; Stewart/Gosain 2006). Das Bild des Softwareentwicklers wird in diesem Rahmen als intrinsisch motivierte Person konstruiert, die sich aus individuellem Interesse und Spaß an der technischen Problemlösung selbstbestimmt engagiert und seine Fähigkeiten in einer entsprechenden Peer Group beweisen will. Die zentrale motivationale Relevanz umfasst damit sowohl den Aspekt des individuellen Vergnügens als auch des Willens zur technischen Perfektion (Feller/Fitzgerald 2000; Sebald 2008).

Der freie Zugang zum bestehenden Wissenskorpus „ist zwar als Arbeitsgrundlage ebenfalls wichtig, wird allerdings nicht so stark betont wie im FSF-Diskurs“ (Sebald 2008: 105) und wird kaum im Hinblick auf soziale, ökonomische oder politische Aspekte hin legitimiert. Die Einbindung von Linux in eine freie Softwarelizenz zeigt zwar ein grundlegendes Bewusstsein für die besondere Relevanz dieses Aspektes, er wird jedoch fast ausschließlich hinsichtlich der kooperativen Praxis und technischen Exzellenz thematisiert.

Eine dritte Semantiklinie, die sich in starker Abgrenzung zur freien Software im Sinne Stallmans reproduziert und sich im Jahr 1997 in Form der Open Source Initiative institutionalisiert hat, verschiebt und erweitert den thematischen Vorrat des freien/offenen Softwarediskurses wiederum erheblich. Sie wird von einigen prominenten Softwareentwicklern vertreten, von denen Eric S. Raymond mit einer Vielzahl grundlegender und bekannter Publikationen der wohl nachhaltigste Repräsentant ist (Sebald 2008). Die Open-Source Initiative sucht aktiv den Anschluss an ökonomische Verwertungsstrukturen und legitimiert diese Zielsetzung mit einer elaborierten und sehr umfassenden Argumentation über die Überlegenheit des praktizierten Kooperations- bzw. Produktionsmodells (Bretthauer 2001).

Die offene Entwicklung von Software wird dabei als marktförmige Ordnung, welche die Koordination der eigennützigen Akteure leistet, konstruiert: Aufgrund seiner Ressourcenüberlegenheit, vor allem im Angebot kreativer Ideen und Problemlösungen sowie mittels der effektiven Koordination über Angebot und Nachfrage, sei das praktizierte Produktionsmodell der hierarchisch organisierten Softwareproduktion überlegen. Seinen prominentesten Ausdruck findet dieses Verständnis in der Metapher des Basars, dem das Bild der monolithisch angelegten Kathedrale entgegengesetzt wird (Raymond 2001). Als besondere Vorteile werden die Fehlerentdeckung und -behebung sowie das flexible Eingehen auf Anwenderbedürfnisse hervorgehoben und als erstrebenswerte Eigenschaft dieser Produktionsweise ausgezeichnet. Technische Qualität bleibt damit auch innerhalb dieser Semantik ein hervorgehobener Wert (Carillo/Gosain 2008; Feller/Fitzgerald 2000; Ljunberg 2000; Stewart/Gosain 2006). Die Argumentation der offenen Softwareentwicklung nimmt nun jedoch ihren Ausgangspunkt vom Kooperations- bzw. Produktionsmodell, dem ein gleichrangiger Eigenwert zugesprochen wird (Bretthauer 2001; Ljunberg 2000; Sebald 2008). In die-

sem Rahmen werden insbesondere der Aufbau einer Community, die gegenseitige Unterstützung und die Möglichkeit zum Lernen betont und als verdienstvolle Leistungen artikuliert (Elliot/Scacchi 2003, 2004, 2008; Lehmann 2004; Stewart/Gosain 2006). Diese Wertvorstellungen erfahren eine zunehmend praxisorientierte Konkretisierung und Normierung im Verbot der Initiierung einer Projektaufspaltung („No Forking“), der Entfernung von Namen aus der Projektliste ohne Einverständnis der Person und der Verbreitung von Code-Änderungen unter Nichtberücksichtigung bestehender Kanäle bzw. Administratoren (Carillo/Okoli 2008; Ljungberg 2000; Stewart/Gosain 2006). Geboten ist zudem eine ausführliche und zugängliche Dokumentation und die grundsätzliche Akzeptanz von „Neulingen“ (Elliott/Scacchi 2003, 2004, 2008).

Die Definition offener Software übernimmt im Wesentlichen die Kriterien freier Software, erleichtert in ihren Lizenzformen jedoch die Integration in proprietäre Software. Denn die Durchsetzung des offenen Softwaremodells benötige Investoren, Sponsoren und Serviceunternehmen, um schädliche Monopolstellungen – namentlich Microsoft – am Markt zu brechen. Der freie Zugang zum Quellcode wird dabei als Voraussetzung für die offene Produktionsweise von Software angesehen – der Bezug zu ethischen, moralischen oder (sozial-)politischen Aspekten jedoch grundlegend abgelehnt (Sebald 2008).

Zusammenfassend lässt sich attestieren, dass der Diskurs der freien/offenen Softwareentwicklung mit seinen drei großen Semantiklinien einige wesentliche (gemeinsame) Grundlagen aufweist. Dazu zählen die Überzeugung von einem frei zugänglichen Wissenspool und die Vorstellung vom frei und selbstbestimmt arbeitenden Programmierer (Sebald 2008). Während sich jedoch die erste Semantiklinie einer moralischen, (sozial-)politischen Argumentation bedient, um primär den freien Zugang zu Wissen bzw. zum Quellcode zu sichern, fokussiert der Diskurs um das Betriebssystem Linux den Aspekt der technischen Qualität und ihrer kooperativen Realisationsmöglichkeiten in einer völlig unpolitischen Semantik. Programmierer streben in dieser Weltanschauung aus persönlichem Vergnügen nach technischer Exzellenz und suchen diese im gemeinsamen Zusammenschluss sowie durch Beitragskombination zu verwirklichen. Als dritte Semantiklinie knüpft die Open Source Initiative in ihrer diskursiven Praxis an das technikorientierte Kooperationsmodell der „Linux“-Generation an und entwickelt es in Anschluss an neoliberale Marktvorstellungen zu einem elaborierten Produktionsmodell der Softwareentwicklung.

2.4 Die Konstruktion des „Users“

Die Rolle der User im Bereich der freien/offenen Softwareentwicklung ist stark von ideologischen Prämissen geprägt. Die Idealvorstellung ist die Aufhebung der Differenz von Entwickler und Laien-Anwender, die mit der fortschreitenden Spezialisierung der Informatik einen zunehmend utopischen Charakter aufweist. Die Ermächtigung der Anwender im Rahmen des Entwicklungsprozesses bleibt jedoch eine stark in der Gemeinde verankerte Überzeugung, die als wesentliches Abgrenzungsmerkmal zur proprietären Softwarebranche formuliert wird. Prinzipielle Offenheit und Hilfsbereitschaft gegenüber Akteuren, die sich beteiligen möchten, werden durch diese Idee mitgetragen (Elliot/Scacchi 2008; Iivari 2008).

In der Entwicklungspraxis werden dem aktiven User[5] ein Reihe funktionaler Eigenschaften zugeschrieben, welche die Rollenstruktur – primär nach Programmierfähigkeiten bzw. technologischem Verständnis – differenzieren. Wichtigste Funktion, die dem Anwender der Software zugeschrieben wird, ist das Testen der Softwarestabilität und die Meldung von Softwarefehlern (Bugs) (Iivari 2008). Dies kommt am deutlichsten im sogenannten „Linux Law“ zum Ausdruck: „Given enough eyeballs, all bugs are shallow“ (Raymond 2001: 30). Dahinter steht die Überzeugung, dass eine große Anzahl von Beta-Testern durch schnelle Fehleridentifizierung und –charakterisierung deren Beseitigung erleichtert und eine große Stabilität der Software sichert (Raymond 2001). Zudem wird die positive Bedeutung des Users als Rückmeldeinstanz betont, die Informationen über die Bedürfnisse der Anwender (Feature-/Support-Request) bereitstellt und als Ausgangspunkt für die Entwicklung neuer Ideen dient (Iivari 2008). Dem aktiven User wird dabei eine generelle Eigeninitiative unterstellt, die sowohl das selbstständige Studium der Software bzw. des Quelltextes, eigenständiges Lernen und die aktive Übernahme programmierferner Aufgaben (Dokumentation, How-to-Guide etc.) umfasst (Iivari 2008).

2.5 Rechtliche und ökonomische Rahmenbedingungen

Die urheberrechtliche Schutzfähigkeit von Software war lange Zeit umstritten und wurde erst Anfang der 1990er Jahre – in den verschiedenen Rechtsräumen zu leicht unterschiedlichen Zeitpunkten – umfassend etabliert (Lemley 1995). Prinzipiell ist damit jeder Entwickler urheberrechtlicher Rechtsinhaber seiner Erzeugnisse und genießt einen weitgehenden Schutzumfang, der aufgrund der leichten Verbreitungsmöglichkeiten von Software die Weitergabe, Vervielfältigung und Bearbeitung einschließt. So erfüllt das Laden eines Programmes in den Arbeitsspeicher – der technische Vorgang der Benutzung einer Software – schon den Tatbestand der Vervielfäl-

[5] Als aktive User werden all jene Akteure bezeichnet, die im Gegensatz zu den Entwicklern keine grundlegenden Beiträge zum Quellcode der Software beisteuern, jedoch im Austausch mit diesen stehen und nicht, wie passive User, die Software lediglich nutzen (Crowston/Howison 2005).

tigung und bedarf der urheberrechtlichen Erlaubnis (Jaeger/Metzger 2002). All diese Aspekte lassen sich mit Lizenzregelungen kostenpflichtig übertragen und ökonomisch verwerten, wie es bei proprietärer Software üblich ist.

In der freien/offenen Softwareentwicklung gelten dieselben urheberrechtlichen Bedingungen, die sich jedoch aufgrund der Kooperation vieler Programmierer in verschiedenen Konstellationen darstellen: (1) urheberrechtliche Einzelwerke, die zu einem verbundenen Werk zusammengefügt werden, (2) Miturheberschaft im Rahmen eines gemeinsamen Konzeptes sowie (3) die Ableitung bzw. Bearbeitung eines bestehenden Werkes (Jaeger/Metzger 2002). Zur Erhebung von Ansprüchen bei rechtlichen Verletzungen müssen dabei alle Miturheber namentlich bekannt sein, während Unterlassungsansprüche vom Einzelnen für alle Beteiligten geltend gemacht werden können (Jaeger/Metzger 2002). Letzteres eröffnet eine recht einfache Möglichkeit der Rechtsdurchsetzung, da nur ein urheberrechtlich Betroffener für ein Verfahren notwendig ist.

Freie/offene Softwarelizenzen[6] nutzen den bestehenden urheberrechtlichen Schutz zur Durchsetzung ihrer Ziele, indem der Urheber mittels der Lizenz den Nutzern das Recht auf Benutzung, Untersuchung und Veränderung der Software sowie deren Vervielfältigung und Verbreitung erlaubt (Jaeger/Metzger 2002). Dabei greifen sie auf das sogenannte Copyleft-Prinzip zurück, welches diese Rechte unter die Bedingung der Beibehaltung der Lizenz stellt und so die Transformation freier/offener Software in proprietäre Software unterbindet (Jaeger/Metzger 2002). Die Veröffentlichung und Verbreitung sowohl der ursprünglichen als auch der veränderten bzw. weiterentwickelten Software muss also stets unter dem gleichen Lizenztyp fortgesetzt werden, unter den die Software erstmalig gestellt wurde. Für Vervielfältigung und Verbreitung seitens des Nutzers dürfen keine Lizenzgebühren verlangt werden, lediglich ein einmaliger Kaufpreis für die Software, Softwarezusammenstellungen oder zusätzliche Serviceleistungen darf eingefordert werden (Jaeger/Metzger 2002). Es ist unschwer zu erkennen, dass unter diesen Bedingungen das Lizenzgebührenverbot leicht durchzusetzen ist, da die kostenfreie Vervielfältigung und Verbreitung des Programms seitens der Nutzer erlaubt und über das Internet leicht realisierbar ist. Dieser Umstand führt auch dazu, dass mit Ausnahme einiger großer Softwaredistributionen für Unternehmen mit entsprechendem Betreuungsservice die überwiegende Anzahl freier/offener Software im privaten Anwenderbereich grundsätzlich kostenfrei angeboten wird.

[6] Im Folgenden werden exemplarisch die grundlegenden rechtlichen Rahmenbedingungen der freien *und* offenen Softwarelizenzen anhand der GNU-Lizenz (General Public Licence) dargestellt, die als Grundtyp freier/offener Lizensierung und als eine der am weitesten verbreiteten Lizenzformen eine hohe Bedeutung hat (Jaeger/Metzger 2002).

2.6 Zwischenfazit

Die freie/offene Softwareentwicklung lässt sich auf einer gesellschaftstheoretischen Ebene in ihren Anfängen als soziale Bewegung fassen – ein Begriff, der in der Systemtheorie einer abschließenden Definition entbehrt und eng mit dem Konzept der Protestbewegung verbunden ist. Dieses soziale Phänomen lässt sich als eigene Systemform unter den Bedingungen der Autopoiesis und Abgrenzung zu einer Umwelt konzipieren, welche im Protest seine grundlegende Form findet und in eine gewisse Distanz zur Gesellschaft bzw. zu gesellschaftlichen Zentren tritt. Die operative Schließung sozialer Bewegung realisiert sich über das Protestthema:

> „Wenn man sich an dem Protestbegriff orientiert, kann man Einheiten, soziale Einheiten, d. h. Kommunikationsmengen herausgreifen, die sich selber von der Umwelt abgrenzen, indem sie sich bestimmte Protestthemen herausgreifen und diese kommunikativ behandeln, so daß eine Kommunikation als zugehörig oder nicht zugehörig erkennbar ist“ (Luhmann 1994: 176).

Das Thema des Protests dient in diesem Sinne als Leitdifferenz: Auf der Grundlage der (Selbst-)Beobachtung werden gegebene Situationen in Form des Widerspruchs thematisiert und Realitäten anderer Art als Gegenentwürfe kommuniziert – ein Phänomen, das sich als „das Ziehen einer Grenze *in* einer Einheit *gegen* die Einheit“ (Luhmann 1995a: 201) formulieren lässt.

Die freie/offene Softwareentwicklung konstituiert sich in diesem Sinne als eigene Einheit in Abgrenzung zur proprietären Softwarebranche durch Vernetzung von Kommunikationen, die sich mit einem abgrenzbaren Protestthema befassen (Luhmann 1994). Die offene Thematisierung bzw. Konstruktion von Dysfunktionen der Entwicklungen im Bereich der Softwareproduktion gleichsam „von außen“ stellt dabei die Grundlage für einen eigenen Gegenentwurf dar: Zunächst zielt der Protest auf die Bewahrung einer bestehenden Praxis der Softwareentwicklung, doch entwickelt sich im diskursiven Verlauf eine zunehmend eigenständige und angereicherte Vorstellung von freier/offener Softwareentwicklung und deren Bedeutung(en). Für soziale Bewegungen prägend ist nach Luhmann in diesem Zusammenhang eine moralisch und emotional[7] beladene Kommunikation, die nicht zuletzt auf die Mitgliedschaftsmotivation durch Anschlussmöglichkeiten für Motive, Commitments und Bindungen zielt (Luhmann 1994). Stallmans politische Argumentation mit Verweis auf die Grundrechte, Selbstverwirklichungs- und Bildungschancen entspricht dem sehr deutlich; vor diesem Hintergrund fokussiert seine Kritik primär soziale Dysfunktionalitäten der proprietären Distributionspraxis.

[7] Nicht im Sinne psychischer Zustände (die natürlich vorliegen können), sondern als Thema und Form der Kommunikation.

Das Phänomen der freien/offenen Softwareentwicklung weist jedoch einige Merkmale auf, die sich mit einem systemtheoretischen Verständnis von Protestbewegungen nicht erschöpfend beschreiben lassen und entsprechender Anpassung bedürfen. Erfasst man soziale Bewegungen mit Luhmann als soziale Einheiten, die öffentlich Protest kommunizieren, aber die Lösung der Probleme in den Zentren der Funktionssysteme verorten, liegt deren Leistung „lediglich" in der Bereitstellung von (Selbst-)Beschreibungen (Luhmann 1997, 1986b). Die diskursive Praxis der freien/offenen Softwareentwicklung bildet und institutionalisiert jedoch Strukturen, die neben dem Protestaspekt auch eine „Anti-*Praxis*" umfassen. Sinn- und Erwartungsstrukturen, die als solche das System der freien/offenen Softwareentwicklung abgrenzen und konstituieren, weisen eine deutliche Ausrichtung auf alternative Produktionsprozesse der Softwareentwicklung und deren Vollzug auf[8]. Eine entsprechende Handlungspraxis begleitet den Diskurs der freien/offenen Softwareentwicklung von Beginn an, und die Schaffung von Softwarelizenzen sichert die Reproduktion dieser Praxis auch rechtlich. Als ‚reine' Protestbewegung lässt sich diese Bewegung aus einer systemtheoretischen Perspektive daher nicht erfassen.

Die Sinn- und Erwartungsstrukturen der freien/offenen Softwareentwicklung sind sachlich vor allem über relativ abstrakte Identifikationen, primär Werteideale, bestimmt. Über eine Reihe an Programmen und auch Rollenkonstruktionen konstituieren bzw. spezifizieren sich als solche auch konkretere (Handlungs-)Erwartungen und realisieren in diesem Maße auch Wertbestimmungen. Allerdings verbleibt der allgemeine Sinnrahmen auf dem recht abstrakten Niveau von Werten, von denen aus Handlungen nur bedingt selektiert bzw. konditioniert werden können (Luhmann 1987). Dieser Aspekt ist für den Bereich der freien/offenen Softwareentwicklung von besonderer Bedeutung, da zwar einerseits ein gewisses Maß an struktureller Unbestimmtheit vorliegt, sich jedoch anderseits auch eine spezifische, soziale Durchsetzungskraft dieser Form ergibt.

Im Verbreitungsmedium Internet, welches das zeitunabhängige Erreichen einer fast unbegrenzten Anzahl von Adressen ermöglicht, stellt sich verstärkt die Frage, welche Kommunikation zur Annahme motivieren kann (Luhmann 1987). Diese Problematik hat insbesondere für den vorliegenden Bereich der freien/offenen Softwareentwicklung eine besondere Relevanz, da sich hier der soziale Kontakt fast ausschließlich

[8] Handlung wird im systemtheoretischen Verständnis über Kommunikation und Attribution konzeptionalisiert und ist als mitlaufende Selbstreferenz bzw. Zurechnung elementarer Bestandteil sozialer Systeme (Luhmann 1987). Der Begriff der (Erwartungs-)Struktur lässt sich dementsprechend auf Kommunikation sowie auf Handlung beziehen und wird im Weiteren – Luhmann folgend – primär „auf Strukturen, die die Handlungen eines sozialen Systems ordnen, also auf die Strukturen dieses Systems selbst" (Luhmann 1987: 382) angewendet (ausführlich dazu Luhmann 1987).

über computervermittelte Schriftkommunikation vollzieht[9]. In funktional differenzierten Gesellschaften wird die Unwahrscheinlichkeit der Selektionsannahme „durch Auflösung des zirkulären Verhältnisses von Selektion und Motivation […] gelöst, und zwar dadurch, daß *die Konditionierung der Selektionen zum Motivationsfaktor gemacht wird*" (Luhmann 1997: 321). Selektion(en) und Motivation werden als symbolisches Medium zusammengefasst, um ihre Annahme als Einheit hinreichend sicher zu generalisieren (Luhmann 1987). Diese – in der Systemtheorie als symbolisch generalisiertes Kommunikationsmedium behandelte – Einheit bietet einen geeigneten Ausgangspunkt für die Analyse der vorliegenden Zusammenhänge.

Werte können selbst nur bedingt als symbolisches Kommunikationsmedium angesehen werden, was sich ursächlich vor allem auf die eingeschränkten Möglichkeiten zur binären Zentralcodierung von Werten zurückführen lässt[10]: Werte werden als Unterstellung bzw. Prämissen in die Kommunikation eingeführt und entziehen sich der begründbaren Annahme oder Ablehnung. Aufgrund fehlender Transitivität vollzieht sich die Bestimmung von Werten auf einem recht abstrakten Niveau, welches sich mit zunehmender Spezifikation am jeweiligen Kontext orientieren muss. Damit sind der symbolischen Generalisierung und der Ausbildung medienspezifischer Funktionssysteme Grenzen gesetzt (Luhmann 1997).

Für die vorliegende Perspektive ist dabei insbesondere die Funktion, die Werte für die Kommunikationszusammenhänge der freien/offenen Softwareentwicklung erfüllen, von herausgehobener Bedeutung. Ausgehend vom Bezugsproblem der Erfahrung doppelter Kontingenz, welche die Fortsetzung sozialer Kontakte auf einer gemeinsamen Basis extrem unwahrscheinlich macht, fungieren Werte „durch die rekursive Verknüpfung entsprechender Unterstellungen im Kommunikationsprozeß" (Luhmann 1997: 341) als Orientierungsrahmen des Handelns. Innerhalb der Zurechnungskonstellationen sind sie daher dem Erleben zuzuschreiben: Nicht Werte sind handlungsabhängig, sondern die Handlungen stehen in einem Abhängigkeitsverhältnis zu den Werten (Luhmann 1997). Im Gegensatz zu ‚voll funktionsfähigen' symbolisch generalisierten Kommunikationsmedien entbehren Werte einer universalen Respezifikation. Ihre Bestimmung realisiert sich über Ideologien und Argumentationen, die eine Direktionsleistung erst im spezifischen Kontext entwickeln: „Sie [die Werte; Anmerk. L. L.] explizieren zwar keine Anwendungsbedingungen, aber sie stehen unter Abwägungsvorbehalt, so daß erst im Einzelfall bestimmt werden kann, was zu ihrer Realisierung geschehen kann" (Luhmann 1987: 342).

[9] Größere Projekte der freien/offenen Softwareentwicklung tendieren mit zunehmender Existenzdauer zwar dazu, auch reale bzw. persönliche Treffen zu organisieren – der Hauptbestandteil der Kommunikation bleibt aber internetvermittelt (Ellrich 2004).

[10] Es lassen sich noch eine Reihe weiterer struktureller Schwierigkeiten anführen, wie Technisierung, symbiotische Symbole, Einschluss des Ausschlusses usw. (Luhmann 1997), die in diesem Rahmen jedoch nicht ausführlich behandelt werden können.

Diese Form der „Koordination“ über Werte leistet damit einen gemeinsamen Grundbezug sinnhafter Orientierung, der den Anschlussbereich möglicher, d. h. sinnvoller Operationen begrenzt. Der abstrakte Charakter von Werten begründet dabei zwar eine grundlegende Abhängigkeit von der jeweiligen Kontextsituation, um spezifische Verhaltensdirektionen zu leisten. Dies ermöglicht es aber auch, einen komplexen Werte*kanon* – wie ihn die freie/offene Softwareentwicklung umfasst – zunächst einmal als nicht hinterfragbare Unterstellungen in die Kommunikation einzuführen und mit einer hohen Annahmewahrscheinlichkeit auszustatten. Diese Form der sozialen Ordnungsleistung entspricht dabei auch den inhaltlichen Prämissen der freien/offenen Softwareentwicklung, die der Freiheit bzw. Selbstbestimmung des Einzelnen einen hohen Stellenwert beimessen; sie ist insofern an die themenspezifischen Commitments der Protestbewegung anschlussfähig.

> „Werte sind das Medium für eine Gemeinsamkeitsunterstellung, die einschränkt, was gesagt und verlangt werden kann, ohne zu determinieren, was getan werden soll“ (Luhmann 1997: 343).

Damit ein solches Wertdenken, d. h. „eine regulative Ordnung von Gesichtspunkten des Vorziehens“ (Luhmann 1999b: 25), im Sinne sozialer Ordnungsleistungen auch Handlungsdirektion erfüllt, bedarf es der (kontextualen) Respezifikation der Werteaspekte. Die diskursive Praxis der freien/offenen Softwareentwicklung mit ihrer ausgeprägten Semantik kann als Ergebnis eines solchen Prozesses in Bezug auf das Anwendungsfeld der Softwareentwicklung angesehen werden. Dieser Zusammenhang ist im Wesentlichen durch eine Ursachen-Wirkungs-Zuschreibung strukturiert, wobei die gewünschten (*bewerteten*) Wirkungen in Form eines Zweck-Mittel-Schemas den Handlungshorizont ordnen (Luhmann 1999b).

> „Der Zweckbegriff bezeichnet diejenigen Wirkungen bzw. den Komplex von Wirkungen, die das Handeln rechtfertigen sollen […]. Die Zwecksetzung besagt, daß der Wert der bezweckten Wirkungen *ungeachtet der Werte oder Unwerte der Nebenwirkungen bzw. der aufgegebenen Wirkungen* anderer Handlungen das Handeln zu begründen vermag. Der Mittelbegriff erfaßt dieselben Wertrelationen von der anderen Seite der benachteiligten Werte aus. Er geht von den Ursachen aus, die zum Erreichen einer bezweckten Wirkung geeignet sind“ (Luhmann 1999b: 44).

Die verschiedenen Semantiklinien des Diskurses der freien/offenen Softwareentwicklung variieren in diesem Sinne über unterschiedliche Wertrelationen die Zweckprogramme, welche als solche die Sinnprämissen dieses Systems konstituieren: Zunächst ist es eine recht abstrakte Vorstellung von Freiheit, die vorrangig über rechtliche Kriterien und die Sicherung der freien Zugänglichkeit des Quellcodes gesucht wird. Aufbauend auf diesem Mittel bzw. diesen Mitteln der Offenheit verfolgt die Linux-Gemeinde das Ziel der technischen Äquivalenz und erweitert bzw. verändert den

Kanon zielführender Mittel um entsprechende Vorstellungen der Kooperation. Die Open Source Initiative hingegen entwickelt diesen Aspekt weiter aus und schließt ihn an ökonomische bzw. markttheoretische Zielsetzungen an. In einer globalen Betrachtung zeigt sich, dass die verschiedenen Semantiklinien vor allem in den präferierten Zwecken differieren, während die Mittel eher aufeinander aufbauen und in diesem Zuge erweitert werden.

Die Zwecksetzung der freien/offenen Softwareentwicklung fungiert dabei „nicht nur als Erwartung eines faktischen Verlaufs, sondern auch als Wertschätzung, die über den lohnenden Einsatz systemeigener Kräfte bestimmt“ (Luhmann 1999b: 188). Sie wird in starkem Maße als Handlungsgrundlage generalisiert und institutionalisiert, die als Selektionsstruktur eine Sinnordnung konstituiert und als „koordinierende Generalisierung“ (Luhmann 1999b: 189) fungiert. Als Identität kondensiert sie durch Wiederholung und generalisiert sie ihre Geltung in sachlicher, zeitlicher und sozialer Sinndimension. „Frei“ bzw. „offen“ kann aus dieser Perspektive als Symbol einer autonomen Sinneinheit verstanden werden, die als strukturierte Komplexität den Rahmen für Systemreproduktion und -differenzierung unter den besonderen Bedingungen computerbasierter Kommunikation vorgibt.

3 Die Entwicklerplattform SourceForge.net

3.1 Die Plattform und ihre Organisationstools

Die Internetplattform SourceForge.net (engl.: Quelltextschmiede) ist ein sog. „Repository“, das Softwareentwickler bei der Erstellung und Verwaltung von quelloffener Software unterstützt. Die Plattform stellt seinen Nutzern kostenlos eine Reihe von Tools zur Verfügung, die der Koordination des Softwareentwicklungsprozesses und der Kommunikation dienen und für Dokumentation und Download entsprechenden Datenspeicher bieten. Die Organisation der Entwickler erfolgt in Softwareprojekten, die auf der Plattform durch eigenständige Portale repräsentiert werden. Der grundlegende Aufbau der Projektportale ist durch die Plattform bestimmt, die angebotenen Tools lassen sich jedoch nach Bedarf einbinden und anpassen. Dieses Angebot nutzen mehr als 2 Millionen Entwickler in über 230.000 Projekten mit täglich insgesamt über 4 Millionen Downloads (SourceForge Documentation 1a, 1b).

Im Folgenden werden die wesentlichen Dienstleistungen der Plattform kurz vorgestellt und erläutert, um sie dann zusammenführend unter dem Aspekt der Virtualität – als einer Besonderheit der computerbasierten Kommunikation im Internet – zu verorten und einem analytischen Verständnis zugänglich zu machen. Auf dieser Grundlage zielt die Artefaktanalyse eines exemplarischen Tools der Softwareprojekte – des Trackersystems – auf die Rekonstruktion (latenter) Sinnstrukturen des vorliegenden organisationalen Zusammenhanges. Beobachtungsgrundlage ist das Projekt „Password Safe“, welches auch Gegenstand der späteren Netzwerkanalyse ist.

Versionsverwaltung (Source Code Management, SCM)

Der Begriff der Versionsverwaltung bezeichnet ein technisches System zur Organisation und Koordination von Programmiertätigkeit, d. h. primär zur Aufzeichnung von Quellcodeänderungen bzw. -beiträgen und zu deren Integration in eine gemeinsame Softwareversion. Versionsverwaltungssysteme erlauben es, eine zentrale, aktuelle Version der Software zu speichern, die zeitgleich von mehreren Entwicklern ausgelesen und bearbeitet werden kann. Deren Änderungen am oder Beiträge zum Code werden in die zentrale Version integriert („einchecken“) bzw. in ihr zusammengeführt („merge“), wobei diese zentrale Version in ihrem Entwicklungsverlauf dokumentiert wird. Stabile Entwicklungsstufen der Software oder einzelner neuer Bestandteile (z. B. zur Behebung eines bestehenden Konflikts oder zum Hinzufügen einer zusätzlichen Funktion) werden dann im File Release System (FRS) eingefügt und zum Download freigegeben (Baerisch 2005).

Forum (Newsgroup)

Ein Forum ist eine Diskussionsplattform, auf der zu verschiedenen (Sub-)Themen (Threads) von Teilnehmern öffentlich Beiträge (Postings) verfasst werden können. Foren ordnen ihre Threads und Postings nach einer hierarchischen Struktur an: Im vorliegenden Fall gibt es zwei Foren („Help“ und „Open Discussion“), innerhalb derer alle bisher angelegten Subthemen gespeichert und mit abnehmender zeitlicher Aktualität von oben nach unten angeordnet werden. Die Anordnung der Beiträge innerhalb der Subthemen erfolgt genau umgekehrt ebenfalls chronologisch, d. h. der zeitlich neueste Beitrag steht zuletzt (Misoch 2006).

Tracker

Tracker (Aufspürer, Verfolger) sind Fallbearbeitungssysteme (Issue Tracking Systems) zum Management von Aufgaben (Tickets), die eine schriftliche Erfassung, Klassifizierung und Dokumentation der einzelnen Anliegen erlauben. Übliche Attributierungen der einzelnen Anliegen sind: Verfasser, Erstellungsdatum, Kurzbeschreibung, Priorität, Beauftragter, Status (Open, Pending, Closed) und inhaltliche Kriterien (z. B. Sachkategorien, Gruppen usw.). Im Rahmen der Softwareprojekte werden die Tracker-Systeme nach verschiedenen thematischen Bereichen differenziert – meist Fehlermeldung (Bug Report), Unterstützungsantrag (Support Request) und Eigenschaftenantrag (Feature Request) – und als Kommunikationstelle zwischen Anwendern und Entwicklern genutzt (SourceForge Documentation 2).

Mailing, News, Statistik

Über die Plattform können Nachrichten des Projekts über ein Newsboard für alle Besucher der Seite zugänglich gemacht oder über Mailinglisten an einen festen Adressatenkreis übermittelt werden. Weitere Informationen über das Projekt werden in Form statistischer Kennwerte, welche die Plattform einheitlich für alle Projekte erhebt, bereitgestellt. Dazu zählen: Anzahl der Postings im Forum, Anzahl der Anliegen (Tickets) im Trackersystem, Anzahl der Transaktionen innerhalb des Source Code Management Tool sowie Anzahl der Hits im Download-Bereich und für die gesamte Projektseite (Traffic) für frei wählbare Zeiträume (SourceForge Documentation 3).

Schreibrechte

Die Projektorganisation baut zentral auf der Vergabe von Schreibrechten durch den oder die Administratoren des Projekts auf, welche die vollständige Kontrolle über alle Rechte bezüglich des Projekts halten. Das Berechtigungssystem setzt die „formale“ Mitgliedschaft für die Vergabe von Schreibbefugnissen voraus, welche auf drei verschiedenen Niveaus vorliegt: (1) „Admin/Manager/Editor“ erhalten volle Schreibrechte für alle Daten und die gesamte Feature/Tools-Verwaltung, (2) „Technicians“ kön-

nen einzelne Aufgaben (primär im Tracker-System) bearbeiten und (3) „Moderatoren“ können (ausschließlich) Nachrichtenbeiträge in Mailinglisten, Foren usw. verwalten (SourceForge Documentation 4). Diese Differenzierung wird meist jedoch nur in großen Projekten aufgegriffen, kleinere Projekte vergeben nicht selten an alle Mitglieder volle Schreibrechte. Der Berechtigungsstatus ist nicht öffentlich einsehbar, die Projektmitglieder werden jedoch meist mit Rollenbezeichnungen (z. B. Developer, Translator, Porter etc.) versehen, die ihre Funktion spezifiziert.

Für registrierte User der Plattform ohne Schreibrechte bzw. eine Projektmitgliedschaft ist es möglich, Beiträge im Tracker-System und den Foren zu verfassen sowie sich in Mailinglisten einzutragen. Sie können zudem den Quellcode einsehen und kopieren, jedoch keine eigenen Änderungen im Versionsverwaltungsprogramm des Projekts vornehmen. Für den Download der Software ist keine Registrierung notwendig (SourceForge Documentation 5).

3.2 Virtuelle Organisation

Computervermittelte Kommunikation über das Internet umfasst eine große Bandbreite möglicher Kommunikationsformen, die sowohl synchrone als auch asynchrone Übermittlung auditiver und visueller Inhalte in verschiedenen Kombinationen zwischen den Endgeräten erlaubt (Beck 2006; Döhring 2003). Die freie/offene Softwareentwicklung greift dabei so gut wie ausschließlich auf den Bereich der visuellen, asynchronen Kommunikation – d. h. der digitalisierten Kommunikationsschrift – zurück, die durch einige strukturelle Eigenschaften gekennzeichnet ist.

Computervermittelte Kommunikation über das Internet lässt „die noch bestehenden räumlichen (also zeitlichen) Beschränkungen der Kommunikation gegen Null tendieren“ (Luhmann 1997: 302). Mitteilung und Empfang können zeitlich auseinandergezogen werden und erweitern dadurch die temporalen Möglichkeiten des Vollzugs der Kommunikation, die sich erst mit dem Verstehen als solchem realisiert. Luhmann geht in einer seiner wenigen Ausführungen zu diesem Thema sogar noch weiter:

> „Sie [computervermittelte Kommunikation; L. L.] ermöglicht es, die Eingabe von Daten in den Computer und das Abrufen von Informationen so weit zu trennen, daß keinerlei Identität mehr entsteht. Im Zusammenhang mit Kommunikation heißt dies, daß die Einheit von Mitteilung und Verstehen aufgegeben wird. Wer etwas eingibt, weiß nicht […], was auf der anderen Seite entnommen wird. […] Und ebenso wenig muß der Empfänger wissen, ob etwas und was ihm mitgeteilt werden sollte. Das heißt: die *Autorität* der Quelle mit all den erforderlichen sozialstrukturellen Absicherungen (Schichtung, Reputation) wird entbehrlich, ja durch Technik annulliert und ersetzt durch *Unbekanntheit* der Quelle. Ebenso entfällt die Möglichkeit, die *Absicht* einer Mitteilung zu erkennen und daraus Verdacht zu

nähren oder sonstige Schlüsse zu ziehen, die zur Annahme bzw. Ablehnung der Kommunikation führen könnte“ (Luhmann 1997: 309).

Die skizzierten Entgrenzungen in zeitlicher, sozialer und sogar sachlicher Dimension stellen, wie Luhmann selbst betont, einen Möglichkeitsraum dar – dessen konkrete Entwicklung kaum zu prognostizieren sei (Luhmann 1997). In Bezug auf den schriftlichen Kommunikationstypus der freien/offenen Softwareentwicklung im Internet lassen sich vor diesem Hintergrund einige grundlegende theoretische Prämissen als Rahmenbedingungen – auch für die Systemdifferenzierung – ableiten.

Schriftliche Kommunikation entzieht sich wichtigen Signalen der Kopräsenz, wie Gestik, Mimik, Tonfall usw., und entlastet durch die Entkopplung von Mitteilung und Verstehen von der unmittelbaren Anschlussreaktion in der Interaktion (Luhmann 1997).

> „Zunächst und vor allem wird bei schriftlicher Kommunikation Metakommunikation optional. Sie läuft nicht mehr zwangsläufig mit […]. Textverweise und Kontextverweise (zum Beispiel Verfasser, Absender, Adressaten) müssen explizit eingeführt werden; und es gibt keine soziale Erwartung des unmittelbaren Übergangs zur aktiven Teilnahme, zur Gegenäußerung oder auch nur zur Mitteilung des Verstandenhabens. Deshalb wird die Unterstellung aufgegeben, daß der eigentliche Sinn der Kommunikation in der Metakommunikation liege. Stattdessen erwartet man Information und liest nicht weiter, wenn diese Erwartung allzu unbefriedigend bleibt“ (Luhmann 1997: 257).

Der Mitteilungsvorgang konzentriert sich so in einer reduzierten Form im Text selbst und eröffnet Interpretationsräume, die sehr unterschiedlich ausgefüllt werden können. Der Schwerpunkt der schriftlichen Kommunikation verschiebt sich auf die Information, was die Selektionskoordination zunehmend erschwert, die Kommunikation jedoch auch für indirekte, konnexionistische Anschlüsse erweitert. „Entsprechend gewinnt, verglichen mit der engen Verschmelzung von Reziprozität und Zeit in der mündlichen Kommunikation, die Sachdimension an Bedeutung“ (Luhmann 1997: 276), und das Grundproblem der doppelten Kontingenz sowie entsprechender sozialer Ordnungsleistungen dürfte sich verstärkt in diese Dimension verschieben bzw. Kompensationsleistungen hervorbringen.

Die Computertechnologie erweist sich hier als Medium, welches die Formen der Kommunikation nicht determiniert, „denn dazu gehören die Ereignisse des Eingebens und Entnehmens von Information. Aber die Programme sind, wie einst die grammatischen Regeln der Sprache, Formen, die die Möglichkeiten der strikten Kopplung einschränken und damit ins Unabsehbare ausweiten können“ (Luhmann 1997: 309f.). Damit wird ein Aspekt computervermittelter Kommunikation im Internet angesprochen, der in der Analyse organisationaler Kommunikationsstrukturen eine besondere Bedeutung hat: Computer und Software erlauben als Kommunikations-

medium weitreichende Eingriffs- und Gestaltungsmöglichkeiten im Hinblick auf das Medium selbst, d. h., auf die Möglichkeiten der Formgebung kann selektiv Einfluss genommen werden (Sutter 2008). Zudem stellt der Computer fast unbegrenzte Speicherungsmöglichkeiten zur Verfügung, die eine einfache Dokumentation der Schrift bzw. (Text-)Kommunikation erlauben und eine Art soziales Gedächtnis bereitstellen, auf das Bezug genommen werden kann. Die Kommunikation kann sich so vom (menschlichen) Erinnern entlasten, muss jedoch auch mit den zunehmenden Möglichkeiten des Verweises umgehen (Luhmann 1997). Spezifische Kommunikations- bzw. Ordnungsformen lassen sich dagegen nur bedingt aus dem Medium selbst erklären, sie realisieren sich vielmehr in spezifischen technischen Artefakten sowie entsprechender Aneignungs- und Gebrauchsweisen, die es zu untersuchen gilt.

Ein analytischer Zugang zu den Kommunikationsstrukturen der Softwareprojekte der Plattform sourceforge.net setzt ein konzeptionell-theoretisches Verständnis des Gegenstandes voraus, welches die methodische Erfassung und Beobachtung leitet. Hierfür sind die besonderen visuellen Qualitäten des Computers zu beachten, welche den Begriff der „virtuellen Realität“ geprägt haben. Diese Begrifflichkeit ist in vielerlei Hinsicht – insbesondere in einer ontologischen Diskussion – widersprüchlich und vieldeutig (Münker 2010).

> „Was wirklich ist, ist nicht virtuell, und was virtuell ist, ist laut Definition nicht wirklich. […] Die Eigentümlichkeit dieser Paradoxie ist jedoch, dass sie nicht – wie sie es eigentlich tun ‚sollte‘ – eine Situation der Lähmung und der Erstarrung generiert, sondern einen operativen Raum neuer Art schafft, der sogar die Ebene der Wahrnehmung einbezieht“ (Esposito 1995: 188).

Virtuelle Realität beschreibt – insofern ist ihre begriffliche Widersprüchlichkeit geradezu bezeichnend – eine neue Qualität computerbasierter Kommunikation über das Internet. Die technologischen Potenziale dieser Maschine erlauben die Erzeugung weitreichender visueller, symbolischer Darstellungsformen, in die im Sinne einer Interaktivität mit der Maschine verändernd eingegriffen werden kann (Rammert 1999; Sandbothe 2010). Die visuelle Darstellung von Software und Befehlsabläufen auf dem Bildschirm (Interface) des Computers erzeugt für den Anwender einen eigenständigen Erfahrungs- und Handlungsraum, der für weitreichende Gestaltungsmöglichkeiten und Formbildungen offen ist (Ahrens 2003; Ellrich 2002).

> „Als symbolverarbeitende Maschinen können auf der Ebene der Zeichen virtuelle Realitäten geschaffen werden, in die – entgegen den fiktiven Räumen eines Romans – handelnd eingegriffen werden kann. Die, auf der Ebene der Zeichen konkretisierte, Wirklichkeit im virtuellen Raum ist eine eigenständige Wirklichkeit insofern, dass sie auf nichts weiteres verweist, als auf sich selbst“ (Ahrens 2003: 179).

Die Räumlichkeit dieser Wirklichkeit kann, auch wenn sie natürlich in Form des materiellen Unterbaus der Hardware gegeben ist, nicht im Sinne eines geografischen Verständnisses ausgelegt werden. Die Unterscheidung von Ort und Raum nach Giddens ist dabei ein hilfreicher Ausgangspunkt, auch wenn seine Überlegungen zum „Reembedding" den Kern des betrachteten Phänomens nicht erfassen (Strübing 2006). Virtueller Raum bezeichnet eine „Ordnungsleistung, die durch die Bezugnahme auf den Raum entsteht" (Ahrens 2003: 187), und zwar als konkreter, den Beobachter einschließender (Interface-)Kontext (Esposito 2002). Virtuelle Räume sind als soziale Räume zu verstehen, die (ausschließlich) durch Kommunikation als solche generiert werden und eine spezifische Sinnordnung repräsentieren (Ahrens 2003; Beck 2003).

> „Demnach verweist der Raumbegriff auf das Ordnen, das durch das ‚in-Beziehung-Setzen' verschiedener Kontexte geschieht. In diesem Raumverständnis geht es um das Konstruieren und Herstellen einer bestimmten Ordnung […]. Die Betonung liegt somit auf den spezifischen kulturellen Hervorbringensweisen des Raumes" (Ahrens 2003: 187).

Die Programme und Tools der Plattform sourceforge.net sowie ihr struktureller Aufbau und ihre visuelle Erscheinungsform können aus dieser Perspektive als technische Artefakte betrachtet werden, die konstitutiver Bestandteil sozialer Beziehungen und ihrer Ordnung sind (ausführlich: Rammert 2006). Als virtueller Raum realisieren sie nicht nur einen „neutralen" Funktionsmechanismus, sondern schließen eine symbolisch-kommunikative und praktisch-materielle Wirkdimension ein. Praktiken des Umgangs mit bzw. „durch" das Medium sind nicht einseitig durch dessen technische Aspekte determiniert – ihre Formen sind Ausdruck sozialer Sinnstrukturen, die das technische Artefakt als solches in Wechselwirkung mit einer entsprechenden Praxis (re-)produzieren (Braun-Thürmann 2006).

Die Rekonstruktion geteilter bzw. systememergenter Sinnstrukturen als organisationaler Ordnungsrahmen eines Softwareentwicklungsprojekts bedarf daher „eines analytischen Zugriffs […] auf diese technischen Artefakte und die in ihnen enthaltene ‚stille' Sozialität" (Strübing 2006: 270). Als methodischer Zugang, der über die Analyse „materialisierte[r] Produkte menschlichen Handelns" (Lueger 2010: 92) auf die Ergründung kontextueller Sinnstrukturen zielt, bietet sich die Artefaktanalyse nach Lueger/Froschauer an (Lueger 2010; Froschauer 2009; Froschauer/Lueger 2007):

> „Die interpretative Erschließung von Artefakten unterstellt in der Verweisungsstruktur erst einmal, dass sie Formen von Sinnkristallisationen sind. Als solche entäußern sie aufgrund ihrer Integration in einen Handlungskontext kollektive Sinnstrukturen, die sich im nachvollziehenden Verstehen als Bedeutungen, Handlungsaufforderungen oder Funktionen deuten lassen. […] Als Ausdruck sozialer Sachverhalte im Sinne sozialer Ordnung und gesellschaftlicher Organisierung unterliegen sie mehr oder weniger fixierten Auslegungen von Deutungsge-

meinschaften im Alltagsleben, deren physische und kognitive Praxis in den Artefakten zum Vorschein kommt“ (Lueger 2010: 98).

Die konkrete Materialität des Artefakts spielt dabei eine untergeordnete Rolle, so dass sich mit der Methode ein sehr breites Gegenstandsfeld erschließen lässt (Lueger 2010). Strübing (2006) schlägt das Verfahren daher auch explizit zur Analyse internetbasierter Räumlichkeiten als ethnografischen Zugang einer soziologischen Analyse vor. Aus dieser Perspektive lassen sich Plattform und Projekte als virtueller Raum erfassen, dessen spezifische Gestalt durch die strukturierte, visuelle Einbindung der angebotenen Tools geprägt ist: Dieser Metaphorik folgend stellt die Plattform ein Areal zur Verfügung, auf dem die Projekte mit vorgegebenen Bauelementen (Tools) geschlossene, binnendifferenzierte räumliche Arrangements bilden[11]. Die Tools lassen sich in diesem Sinne als technische Artefakte analysieren und unter den Aspekt sozialer Ordnungsleistung in einen zu (re-)konstruierenden Kontext interpretierend einbinden.

3.3 Artefaktanalyse

3.3.1 Methode und Vorgehen

Die Analyse von Artefakten „als materialisierte Produkte menschlichen Handelns“ (Lueger 2010: 92) richtet sich auf vielfältige und sehr unterschiedliche Untersuchungsgegenstände. Das methodische Vorgehen bedarf daher der Anpassung an das untersuchte Material unter Berücksichtigung des Erkenntnisinteresses, verfügt jedoch über eine allgemeine Systematik. Das Verfahren der Artefaktanalyse folgt einem 5-phasigen Vorgehen der Kontext(re)konstruktion durch Perspektivenvariation, das auf die immanenten Sinn- und Bedeutungsstrukturen des Artefakts zielt (Lueger 2010). In diesem Zusammenhang sind die „Bedingungen des Auftretens, der Herstellung, des Gebrauchs und der Aneignung von Artefakten“ (Lueger 2010: 102) zu berücksichtigen, die den analytischen Zugang zu den kontextualen Strukturen des Artefakts leiten. Die fünf Perspektiven der Artefaktanalyse werden im Folgenden kurz beschrieben, um dann das Vorgehen bei der Analyse des Tracker-Systems zu erläutern.

Die erste Perspektive widmet sich der Ausarbeitung des Forschungskontextes (1), d. h. der Bestimmung des Erkenntnisinteresses innerhalb eines Forschungszusammenhanges in Verbindung mit der Artefaktanalyse. In diesem Schritt wird der umfassende Forschungsrahmen ausgearbeitet und die Relevanz des Artefakts sowie der grundlegende Zugang zum Artefakt abgeleitet. Es folgt die deskriptive Analyse (2) des Artefakts, die dieses als gleichsam fremdes Objekt in seiner Materialität, internen

[11] Einen anschaulichen Beitrag zur sinn(lichen) Orientierung in räumlich, visuellen Arrangement findet sich bei Kerscher (2000).

Differenziertheit und seinen Existenzbedingungen analysiert und detailliert beschreibt. Im Fokus stehen die spezifischen Eigenschaften, d. h. die Komponenten, Relationen und Strukturen des Artefakts und deren allgemeiner sozialer Kontext. Die dritte Perspektive der alltagskontextuellen Sinneinbettung (3) verfolgt die Interpretation des Artefakts aus der Perspektive eines alltagskompetenten Beobachters. Welche Bedeutungsassoziationen werden mit dem Artefakt und seinen Komponenten verbunden, wie lässt sich der Kontext des Artefakts deuten, und können symbolische Bedeutungszuweisungen ausgemacht werden? Die distanziert-strukturelle Analyse (4) richtet ihren Fokus dann zunehmend auf den für das Artefakt relevanten Kontext und dessen Struktur in Form einer sinnhaften Einbettung. Leitende Aspekte sind der Produktionszusammenhang und die potenziellen Wirkungen und Funktionen des Artefakts, der Umgang mit dem Artefakt und seine Rezeption sowie die szenische Analyse im Sinne einer Integration in einen gesellschaftlichen bzw. lebensweltlichen Gesamtzusammenhang. Die Ergebnisse werden in einer vergleichenden Analyse (5) sowohl artefaktintern als auch in Beziehung zu vergleichbaren Artefakten und typischen situativen Kontexten einer globalen Perspektive zugeführt. Dieser letzte Schritt dient dem Erkennen typischer Eigenheiten, spezifischer Kontexte und der charakteristischen Einbettung des Artefakts (ausführlich: Lueger 2010).

Die folgende Analyse orientiert sich in ihrem methodischen Vorgehen an der Systematik der Artefaktanalyse in Organisationen nach Froschauer (2009), da der vorliegende Bereich der kooperativen Softwareentwicklung sich als organisationaler Kontext charakterisieren lässt (Froschauer 2009). Die obigen fünf Perspektiven werden nach Maßgabe der Schlüsselelemente Dekonstruktion und Integration in zwei methodische Schritte unterteilt: Der erste Schritt der „dekonstruktiven Bedeutungsrekonstruktion“ integriert die deskriptive Analyse des Artefakts sowie deren alltagskontextuelle Sinneinbettung. Distanziert-strukturelle und vergleichende Analyse werden im zweiten Schritt der „Rekonstruktion der latenten Strukturen der Organisation“ zusammengeführt (Froschauer 2009). Die Analyse folgt einer grundsätzlich offenen Fragenstellung, der Schwerpunkt des Erkenntnisinteresses liegt jedoch auf der strukturellen Organisation des vorliegenden Kommunikationszusammenhanges in zeitlicher, sachlicher und sozialer Dimension. Tracker-Systeme sind ein Kernelement der virtuellen Räumlichkeit des vorliegenden Softwareprojekts und „als solche entäußern sie aufgrund ihrer Integration in einen Handlungskontext kollektive Sinnstrukturen, die sich im nachvollziehenden Verstehen als Bedeutungen, Handlungsaufforderungen oder Funktionen deuten lassen“ (Lueger 2010: 98). Geteilte Sinn- und Erwartungsstrukturen und darauf aufbauende organisationale Ordnungsleistungen und -anforderungen können so rekonstruiert und als Erkenntnisrahmen für die folgende strukturelle Untersuchung der Projektkommunikation herangezogen werden.

Die Analyse wurde in einem Team durchgeführt, welches aus drei Personen aus verschiedenen Fachrichtungen und unterschiedlichen Geschlechts bestand. Auf eine inhaltliche Interpretation der getätigten Kommunikationen wird aufgrund des spezialisierten Fachjargons verzichtet. Das Tracker-System, hier exemplarisch am Bug-Tracker untersucht, umfasst als spezifischen Kommunikationsraum mehrere Webseiten innerhalb der Projektplattform „Password Safe“, die über Hyperlinks verbunden bzw. aufrufbar sind. Es handelt sich in diesem Sinne um eine Auswahl spezifischer Seiten, die als Gegenstand der Analyse vorgegeben wurden (Anhang I). Diese Seiten wurden in einem ersten Schritt der ausführlichen deskriptiven Beschreibung und Analyse unterzogen, wobei insbesondere die Form, Größe und Farbe einzelner Text- bzw. Linkelemente und das strukturelle Arrangement beobachtbar sind. Im Rahmen der folgenden Interpretation des Artefakts und seiner Komponenten aus einer alltagskompetenten Perspektive standen primär Aspekte der allgemeinen (symbolischen) Bedeutung und möglicher Funktionen einzelner Elemente im Vordergrund. Es wurde jedoch schnell deutlich, dass die Aktivierung von einigen Links und Steuerelementen auf den betrachteten Webseiten für eine fundierte Analyse, gerade auch für spätere Interpretationsschritte (des Kontextes), unumgänglich war. Die entsprechenden Verlinkungen – respektive: die Plattformregistrierung sowie einzelne Beiträge im Tracker-System – wurden in Form offener Beobachtungen in den Interpretationsprozess einbezogen, jedoch nicht jeweils einer detaillierten Deskription unterzogen. Als Eigenheit des Umgangs mit dem Artefakt ist dieses Vorgehen jedoch selbst ein wichtiger Hinweis, der in die Analyse einbezogen werden kann.

Die Rekonstruktion latenter Sinnstrukturen der Organisation bezog sich vor allem auf die Bereiche des Gebrauchs, der Funktionen und der sozialen Bedeutungen, für die sich im Anschluss an das Artefakt weitreichende Interpretationsmöglichkeiten ergaben. Dabei ist insbesondere auch die spezielle Sequenzialität der Webseiten und ihrer Verlinkungen zu beachten, die den virtuellen Aufbau des Beobachtungsgegenstands maßgeblich ausmachen. In einem komparativen Vergleich mit einem ähnlichen Artefakt des Projekts und generellen externen Verwendungszusammenhängen wurde die Analyse abschließend um zwei weitere Dimensionen erweitert. Im Folgenden werden die Ergebnisse der Artefaktanalyse zusammenfassend dargestellt und in ihrer Herleitung erklärt.

3.3.2 Ergebnisse

Die virtuellen Räumlichkeiten des (Bug)Tracking-Systems deuten auf eine allgemeine soziale Offenheit hin, die einen freien und selbstständigen Zugang des Users intendieren. Die Webseiten des Tracking-Systems wie die der Projektplattform in Gänze sind für jeden Akteur mir Internetanschluss einsehbar und weisen eine hohe (sachliche) Strukturierung mit ausgeprägtem Orientierungsangebot an: Die Webseiten sind

durch eine sich wiederholende, am oberen Rand beginnende Rahmung versehen, welche die grundlegenden Bereiche der Plattform und des Projekts in Form von Hyperlinks angeben und zum Aufruf bereitstellen. Der Rahmen folgt seitenabwärts einer Differenzierungslogik mit zunehmender Untergliederung und bietet durch Angabe eines Pfadverzeichnisses über die bisherigen (formalen) Navigationsschritte, großformatige und nicht verlinkte Überschriften sowie visuelle Hervorhebungen der ausgewählten Links eine Vielzahl von Hinweisen zum gegenwärtigen „Aufenthaltsort“ innerhalb der Plattform und ihres Inhalts.

Die spezifischen Komponenten des Tracking-Systems bzw. des jeweils ausgewählten Bereichs folgen unterhalb der Navigationsrahmung. Der User gelangt nach Aufruf des (Bug-)Tracking-Bereichs standardisiert auf eine Übersichtsseite, die ihm in tabellarischer Anordnung mit einer Reihe von Attributierungen in der Kopfzeile die letzten 25 Fehlermeldungen (Artefakte) mit Kurzbeschreibung aufzeigt. Die zeitliche Anordnung und relativ geringe Anzahl der angezeigten Fehlermeldungen weist auf eine schnelle Prozesslichkeit der Verarbeitung und geringe Geltungsdauer der Meldungen hin, deren farblich hervorgehobene und prominente Darstellung auf den Anspruch der Anwenderinformation. Als eine Form sozialen Gedächtnisses kann diese Dokumentation in zweierlei Weise verstanden werden: (1) als Informationsangebot, das dem Anwender Zugang zu zeitlich gesichertem, gemeinsamem Wissen garantiert und ihn vom Erinnern entlastet, und (2) als „schnelle“ (und erzwungene) Information für den Beitragsverfasser, um die Wahrscheinlichkeit einer wiederholten, doppelten Meldung zu reduzieren. Die weiteren Attributs- und Ordnungskriterien der Ergebnistabelle – ID, Status, Assignee, Submitter, Resolution und Priority – sowie Möglichkeiten der Filterung der Ergebnisse decken sowohl soziale, zeitliche und vor allem sachliche Aspekte ab und erlauben recht differenzierte Zugangsweisen, die auf heterogene Anwender mit unterschiedlichen Interessen oder Zielsetzungen deuten bzw. diese ermöglichen (s. u.). Die stark technische Begrifflichkeit der vorwiegend sachlichen Attributierungs- und Filterkriterien setzen jedoch ein recht hohes Verständnis des Anwenders voraus. Auf der betrachteten Seite finden sich dabei keinerlei Erklärungs- oder Hilfshinweise; vorgegebene Auswahlkriterien erfordern hohe sachliche Kenntnisse, auch wenn eine prinzipielle „None“-Kategorie in diesen Fällen vorliegt.

Die Abgabe einer Fehlermeldung wird über eine Eingabemaske realisiert, die einer übersichtlichen Gliederung folgt. Vor der eigentlichen Verschriftlichung der Meldung (Deskription) wird wiederum eine Einordnung in vorgegebene technische Kategorien verlangt, die mit einem Teil der Filterkriterien übereinstimmen. Die Sachlogik setzt sich also in diesem Bereich fort – Erklärungen zu diesen Begriffen lassen sich auch hier nicht finden, jedoch geht der Eingabemaske eine spezielle Aufforderung zur Angabe des benutzten Betriebssystems mit einem einfachen Beispiel voraus. Diese spezifische, vom vorliegenden Projekt verfasste Information hat ein vergleichsweise

geringes technisches Niveau und fungiert als Zusatz- bzw. Erinnerungshinweis, der sich durch eine sachliche Überforderung der Beitragsverfasser plausibilisieren lässt: Dem hohen technischen Niveau der im Tracker-System inhärenten Sachlogik scheint nicht jeder Anwender gewachsen zu sein.

Der Aufruf der Eingabemaske auf der „vorhergehenden" Übersichtseite erfolgt über einen vergleichsweise kleinen, unauffälligen Link und setzt die Registrierung auf der Plattform voraus, in deren Rahmen dem Anwender der Zugang zu einem ausführlichen Informationsangebot über die Plattform, deren Tools und Benutzung bereitgestellt wird. Für die Registrierung sind lediglich eine E-Mail-Adresse und keine weiteren persönlichen Angaben notwendig. Die Akteure treten öffentlich ausschließlich über einen selbst erstellten Nickname (oder Accountkürzel) in Erscheinung, welcher die Erreichbarkeit der jeweiligen Akteure im Sinne einer sozialen Adresse sichert. Die kommunikative Teilnahme wird mit den Bedingungen der Plattformkenntnis, im Sinne eines Auseinandersetzens mit der Plattform, und der direkten Erreichbarkeit durch Adressierbarkeit verbunden. Letztere signalisiert aufgrund ihrer „Unpersönlichkeit" einen starken Informationsbezug – für sinnhafte Anschlussoperationen scheint dies jedoch nicht immer ausreichend zu sein (Stichwort: Rückfragen). Zudem werden Anwender bei Änderungen der von ihnen verfassten Anliegen über diese Adresse automatisch benachrichtigt: eine Form der Reziprozität, die auf Anschlusserwartungen mit Gegenleistungscharakter des Beitragsverfassers deuten[12]. Als Selektionsmechanismus scheinen obige Eigenschaften ein nicht geringes Maß an (geteiltem) Verständnis zu fordern bzw. zu sichern, was insbesondere in einem heterogenen organisationalen Kontext erforderlich ist.

Tracker-Systeme werden von der Plattform angeboten und im Rahmen der einzelnen Projekte inhaltlich nach ihren Vorgaben genutzt. Sie dienen der formalisierten Verfassung und Bearbeitung von verschriftlichten Anliegen sowie deren Dokumentation. In diesem Sinne realisiert das Artefakt die Strukturierung eines Kommunikations- bzw. Handlungsprozesses nach ausgewählten Kriterien. Auffällig im vorliegenden Fall ist dabei das sachlich-technische Primat dieser Strukturierung: Sowohl die Differenzierung der Tracker-Systeme innerhalb des Projekts als auch die Bestimmung bzw. Attributierungen der Tracker-Meldungen folgen überwiegend technischen Kriterien, die häufig vom Projekt vorgegeben sind und nicht erklärt werden. Der Nutzer des Systems muss sich der gegebenen Sachlogik anpassen sowie ein entsprechendes Wissen oder eigenständige Lernanstrengungen zum Verständnis der Begrifflichkeiten vorweisen.

[12] Häufig werden hier entsprechende Veränderungen des Quellcodes vermerkt und mitgeteilt, die für den Anwender – soweit er nicht selbst über entsprechende Informatikfähigkeiten (Kompilieren) verfügt – erst mittelbar in der nächsten Softwareversion zugänglich sind.

Eingriffsmöglichkeiten in das Tracker-System, die über die Erstellung eines Beitrages hinausgehen, haben nur User mit Schreibrechten für das Projekt, d. h. die eingetragenen Projektmitglieder. Dieser Umstand verweist auf eine grundsätzliche soziale bzw. organisationale Differenzierung, vor deren Hintergrund sich die Funktion der Tracker-Systeme erschließen lässt. Alle durch das Tracker-System vorgetragenen Anliegen können ausschließlich durch Projektmitglieder bearbeitet werden, d. h., nur sie können das Anliegen in Programmiersprache übersetzen und in den Quellcode einfügen oder anderweitig „beantworten“[13]. Die prinzipielle Offenheit im Sinne von Inklusion (s. o.) findet hier eine Form der organisationalen – allgemeiner: systemeigenen – Koordination: in Programmierer- und (gehobener) Anwenderrolle. Thematischer Fokus des jeweiligen Tracker-Systems (Bug, Feature Request und Support Request) sowie seine technisch-sachliche Strukturierung konstituieren bzw. sind Ausdruck geteilter (Kommunikations-)Erwartungen, die sinnhafte Anschlussfähigkeit durch Reduktion der Freiheitsgrade ermöglichen oder wahrscheinlicher machen. Einerseits markiert das Artefakt eine deutliche Grenzziehung innerhalb des Systems der Softwareentwicklung zwischen befähigten bzw. befugten Softwareproduzenten (Leistungsrolle) und gehobenen Anwendern (Publikumsrolle), anderseits fungiert es gerade als „Grenzöffnung“, welche die Aufnahme von Irritationen (Anwenderbeobachtungen) strukturiert ermöglicht und koordiniert.[14]

3.3.3 Komparative Analyse

Vergleicht man das Tracker-System mit dem Kommunikationstool Forum (s. o.), lassen sich einige strukturelle Ähnlichkeiten und Unterschiede erkennen. Auch die Foren werden sachlich nach thematischen Schwerpunkt geordnet und innerhalb dieses Themenrahmens – hier: „Help“ und „Open Discussion“ – weiter in Form von Threads differenziert, in denen eine unbegrenzte Anzahl von Beiträgen verfasst werden kann. Die visuelle Anordnung von Threads und Beiträgen folgt jedoch einer anderen zeitlichen Ordnung: Während Threads mit zunehmender Zeitentfernung der letzten *Aktivität* in der Anordnung nach unten rücken, ist die Beitragsreihung innerhalb der Threads genau umgekehrt. Das deutet auf eine prinzipiell unbegrenzte Geltungsdauer der einzelnen Threads hin, deren Inhalt seitens der User chronologisch gelesen werden sollte. Argumentations- und Kommunikationsverlauf scheinen in Foren dementsprechend eine höhere Bedeutung zu haben – ihr Inhalt lässt sich nicht durch vielfältige Kriterien wie im Tracker-System bestimmen. Die Reduktion auf einzelne Anliegen sowie deren technische und sachliche Klassifizierung und zeitliche Anord-

[13] Auch Projektmitglieder melden gelegentlich Fehler, die sie dann häufig selbst bearbeiten. Das Artefakt dient dann primär der öffentlichen Strukturierung des Koordinationsprozesses Dokumentation.
[14] Auch der „formale“ Eintritt in ein Projekt erfolgt häufig über diesen Weg: Der Anwender zeigt ein aktives Interesse, wird zunehmend mit Aufgaben betraut (im Sinne eines Beweises seiner Befähigung oder des Erlernens entsprechender Fähigkeiten) und wird dann ggf. als Vollmitglied aufgenommen. Für eine ausführliche Fallstudie siehe Ducheneaut 2005.

nung im Tracker-System zeigen im Vergleich dazu eine begrenzte Geltungsdauer und ausgeprägte Ausrichtung auf Einzelanliegen, deren Realisierung mit einer Ausdifferenzierung von Leistungs- und Anwenderrollen einhergeht.

Im Rahmen des Kundenmanagements von Organisationen finden Tracker-Systeme auch außerhalb der freien/offenen Softwareentwicklung Verwendung. Hier dient das System zur Bearbeitung von Kundenanfragen und wird in recht ähnlicher Weise genutzt. Die wohl signifikantesten Unterschiede liegen zum einen in der öffentlichen Dokumentation der Anliegen als auch in den vergleichsweise recht hohen Anforderungen an den Beitragsverfasser im vorliegenden Bereich. Leistungs- und Publikumsrollen sind hier deutlich weniger differenziert: Der Anwender trägt sein Anliegen *selbst* in das System ein und muss entsprechende Eigenleistungen zur Orientierung und zum Verständnis des System und der technischen Attribute aufbringen. Die öffentliche Dokumentation der Anliegen zeugt zudem vom „kollektiven" Charakter der Softwareentwicklung, da sie jedem Interessierten Einblick in das Projekt und die anfallenden Aufgaben bzw. Probleme bietet. Das bedeutet auch, dass die Programmierer in diesem Sinne nicht primär eine (unbezahlte) Serviceleistung für einen Kunden erbringen, sondern gemeinsam mit dem Anwender an der Optimierung der Software arbeiten.

3.4 Zwischenfazit

Kommunikation als elementare Operation sozialer Systeme ist in ihrer computervermittelten (schriftlichen) Form im Internet einer grundlegenden Unwahrscheinlichkeit ausgesetzt. Kommunikation – d. h. Mitteilung, Information und Verstehen (Luhmann 1987) – ist hier räumlich sowie zeitlich ungebunden und weist eine potenziell hohe soziale Reichweite (Verbreitung) auf. Der Verzicht „auf die zeitliche und interaktionelle Garantie der Einheit der kommunikativen Operation" (Luhmann 1997: 258) erfordert jedoch in Bezug auf ein sinnhaftes Verstehen und die kommunikative Annahme entsprechende Kompensationsleistungen mit besonderem Gewicht auf der Sachdimension (s. o.).

Soziale Koordination unter diesen Rahmenbedingungen bedarf einer genaueren Analyse eben dieses Rahmens: des internetverbundenen Computers. Dessen spezifische Eigenheit liegt in seiner Funktion als Medium *und* Maschine: Während er als Verbreitungsmedium die Kommunikation in einem neuen Maße in ihrer Einheit entgrenzt, ermöglicht er als Maschine, die Kommunikation selbst zu strukturieren (Esposito 1993). Diese Eigenschaften finden in virtuellen (Kommunikations-) Räumen ihren Ausdruck, indem sie Kommunikationsmöglichkeiten sowohl nach Inhalt als auch in ihrer Form vorselektieren. Hier zeigt sich, wie erwartet, eine ausgeprägte sachliche Strukturierung, die jedoch durch zeitliche und soziale Aspekte – auch

im Sinne sozialer Begrenzung[15] durch hohe Wissensvoraussetzungen, also sachliche Kriterien – ergänzt wird. Als solcher repräsentiert der virtuelle Raum soziale Sinn- und Erwartungsstrukturen, welche die Kommunikationen als soziales System leiten bzw. ermöglichen und abgrenzen. Das Kommunikationsmedium Computer leistet insofern weitreichende Kopplungs- und Verbreitungsqualitäten durch selektive Strukturierung.

Die Auflösung der Mitteilung als Bezugspunkt des sinnhaften Verstehens zeigt im organisationalen Zusammenhang der Softwareentwicklung zudem eine besondere Auffälligkeit. Das Verstehen bezieht sich primär auf die Information im Sinne der Selbstreferenz: „Diejenige Interpretation ist korrekt, die für den, der die Kommunikation versteht, Sinn hat" (Esposito 1993: 352). Im vorliegenden Fall entspricht dies der „Übersetzungsleistung", die der Entwickler in Bezug auf die Information als „In-Beziehung-Setzen" zur formallogischen Ebene des Quellcodes zu leisten hat. Die Mitteilungsabsichten können in diesem Rahmen als generalisiert betrachtet werden; die sachliche Differenzierung der virtuellen Projektbereiche (hier: Bug-, Support-, Feature-Tracker) und der mitgeteilten Information erleichtert die sachorientierte Anschlussfähigkeit des Entwicklers[16]. Auch die zwingende „anonyme" Registrierung des Users mit lediglich einer E-Mail-Adresse ist aus dieser Perspektive als soziale Anschlussmöglichkeit zur Informationsbeschaffung zu werten. Die potenzielle Inklusion jedes Interessierten bzw. Anwenders in die Entwicklung der Software dürfte gerade auch durch diese Charakteristika der virtuellen Kommunikation realisierbar sein.

Die „Chance der sozialen Berücksichtigung von Personen" (Luhmann 1997: 620) organisiert sich in diesem Zusammenhang in der Differenz zwischen Anwender bzw. aktivem User und Entwickler im Sinne von Publikums- und Leistungsrolle. Die basale Voraussetzung ist die Verfügbarkeit eines Computers mit Internetanschluss, jedoch zeigen die obigen Analysen auch einen gewissen Anspruch an die Kenntnis technischer Begrifflichkeiten und deren Verständnis. Dem folgend ist es präziser, von einer gehobenen Anwenderrolle zu sprechen. Die Organisation der Entwickler zeigt in einigen Aspekten formale Qualitäten: Die Mitgliedschaft wird durch sichtbare Zugehörigkeitssymbole angezeigt und steht „als Symbol für eine besondere Rolle mit bestimmten Rechten und Pflichten […], d. h. als abgesonderter Komplex von Verhaltenserwartungen" (Luhmann 1999a: 35). Diese Verhaltenserwartungen leiten sich hier als Selbstverpflichtung aus dem allgemeinen Wertekanon der freien/offenen Softwareentwicklung ab und sichern als solche im Sinne rudimentärer Erfolgsmedien die prin-

[15] Eine hohe soziale Selektion besteht natürlich schon davor: Die Anwendung der Software und ein allgemeines Grundwissen über die Idee der Open-Source-Entwicklung müssen vorausgesetzt werden.
[16] Interessant sind in diesem Zusammenhang die Unterschiede zwischen Trackern und Foren: Letztere zeigen eine deutlich offenere Strukturierung und zeitlich chronologische Anordnung der Beiträge, die der Mitteilungsebene (vor allem durch deren Verschriftlichung und Emoticons) größeres Gewicht beimessen. Der primäre Bezug auf die Information bleibt jedoch erhalten.

zipielle Annahme von Kommunikation seitens der Mitglieder. Die individuellen Gründe und Motivationen[17] der Teilnahme sind in diesem Rahmen von untergeordneter Bedeutung – für die Organisation der operationalen Zusammenhänge sind geteilte Sinnprämissen und Zwecksetzungen von maßgeblicher Relevanz.

> „Durch Übernahme einer Mitgliedschaftsrolle erklärt sich eine Person bereit, in bestimmten Grenzen Systemerwartungen zu erfüllen. Das System kann dann mit dieser Bereitschaft rechnen, ohne sie von Fall zu Fall ermitteln und motivieren zu müssen. Es kann deshalb die Abwicklung systeminterner Handlungszusammenhänge an rein sachlichen Gesichtspunkten ausrichten und die Handlungsbereitschaft unterstellen, ohne die persönlichen Gründe für die Rollenübernahme jeweils erneut prüfen zu müssen“ (Luhmann 1999a: 42).

Durch den oder die Administratoren besteht dabei ein Mindestmaß an Führungsstruktur, die über Mitgliedschaft bzw. Schreibrechte – auch als Mittel der Sanktion, wie für die formale Organisation kennzeichnend – entscheidet (Luhmann 1999a). Die mit der Mitgliedschaft verbundenen Erwartungen verbleiben jedoch auf einem recht unspezifischen Niveau. Ausgehend von allgemeinen Wertvorstellungen und Zwecksetzungen werden, wie in Kapitel 2 ausgeführt, nur begrenzt konkrete Verhaltensvorgaben formulieren: „Sie stehen unter Abwägungsvorbehalt, so dass erst im Einzelfall bestimmt werden kann, was zu ihrer Realisierung geschehen kann“ (Luhmann 1997: 342). Erwartungen müssen in diesem Rahmen von erwünschten Zuständen bzw. Zwecken abgeleitet werden und liegen nur sehr selten als detaillierte Vorschriften vor. Erschwerend kommt hinzu, dass die Software frei zugänglich ist und nicht den Bedingungen der Knappheit unterliegt, d. h. kaum ökonomische Verwertungsmöglichkeiten aufweist. Die Attraktivität der Mitgliedschaft kann also nicht grundsätzlich durch Geld gesichert werden und ist auf die Befriedigung individueller Bedürfnisse angewiesen. Der „Formalität“, im Sinne einer unter Bedingungen gewährten Mitgliedschaft, sind daher deutliche Grenzen aufgewiesen. Als „graduelle Charakterisierung“ (Luhmann 1999a: 38) ist den hier behandelten sozialen Kooperationszusammenhängen nur ein geringes Maß an Formalität zu attestieren, auch wenn sie eine grundlegende Form der Systemdifferenzierung bilden.

In diesem sozialen Kooperationsrahmen, der geprägt ist durch abstrakte Wertgesichtspunkte der gemeinsamen Entwicklung von Software, strukturieren die Organisationstools den Kommunikationszusammenhang entscheidend: Durch sachliche Differenzierung der unterschiedlichen Tools und ihre inhärente Gebrauchslogik leisten sie eine Selektion der relevanten Mittel und Gesichtspunkte, die als gemeinsame Arbeitsgrundlage die Verwirklichung der offenen Softwareentwicklungsideale (als bezweckte Wirkung) realisieren. In Rückgriff auf die Ergebnisse der Artefaktanalyse las-

[17] Die bestehende, sehr ausführliche Forschung weist auf recht heterogene Motivgründe hin (Übersicht: Luthiger 2004).

sen sich die Tools als „koordinierende Generalisierung“ (Luhmann 1999b: 189) auffassen, die die Möglichkeiten und Formen der Kommunikation sowohl in Bezug auf die Teilnahme, den Themenvorrat und die Mitteilungsform als auch die Zielsetzung vorgeben. „Das System gewinnt auf dem Bildschirm seiner Zwecke für das fragliche Verhalten ein stark vereinfachtes Umweltbild und eine Kooperationsgrundlage, die rasche Verständigung gestattet“ (Luhmann 1999b: 192). In diesem Sinne ist den Organisationstools primär eine soziale Ordnungsleistung durch Selektion eines „Möglichkeitsraumes“ zuzuschreiben, der die operative Grundlage für die Systemprozesse des Projekts darstellt.

Die Projekte der Softwareentwicklung respektive Sinn- und Erwartungsstrukturen konstituieren sich im Kontext einer gesellschaftlichen Semantik der freien/offenen Software, die eine Koordination von Selektionen im Sinne geteilter bzw. emergenter Relevanzstrukturen ermöglicht und realisiert. Vor diesem Hintergrund ordnen die Systeme ihre operativen Zusammenhänge in Bezug auf ihren Organisationszweck – der Entwicklung von Software – in Orientierung an Werten und Sinnprämissen, die als Direktive für opportune Mittel bzw. Formen dienen (Luhmann 1999b). Maßgeblich ist dabei das Ideal der Kooperation und Inklusion – die Differenzierung zwischen gehobener Anwender- und Entwicklerrolle ist ihre funktionale Realisierung. Durch das Verbreitungsmedium Computer verlagert sich der Schwerpunkt der Kommunikation auf die Information und eröffnet sachorientierte, von der Person abstrahierte Anschlussmöglichkeiten, auf welche das System zu seiner (Re-)Produktion zurückgreifen kann. Die virtuellen Räume fungieren dabei als strukturierende „Anlauf- und Verbreitungspunkte in einem Netzwerk von Kommunikationen, das für sich als soziales System geschlossen ist“ (Kuhm 2003: 106).

Im Folgenden soll dem operativen Vollzug der Prozesse der System(re)produktion und -differenzierung – in Erweiterung um einen quantitativen Zugang – nachgegangen werden. In Rückgriff auf die Methode der sozialen Netzwerkanalyse geht es darum, exemplarisch anhand des Softwareprojekts „Password Safe“ die spezifischen Konfigurationen der Selbstorganisation zu erfassen und aus einer funktionalen Perspektive zu interpretieren.

4 Soziale Netzwerke

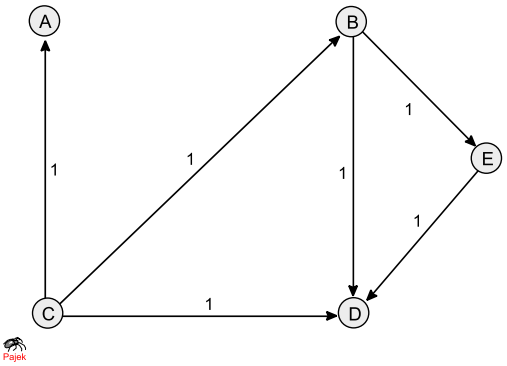
4.1 Das Konzept Netzwerk

Das Konzept des sozialen Netzwerkes basiert auf einem grundlegend relationalen Verständnis sozialer Phänomene. Methodischer Ansatzpunkt und elementare Erklärungseinheit bilden Beziehungen zwischen sozialen Akteuren, deren Qualität eine prinzipiell empirische Größe ist, sowohl in Bezug auf die Beziehungsform (z. B. Kommunikation, Freundschaft etc.) als auch im Hinblick auf entsprechende Quantitäten (z. B. Mitteilungen, Intensität etc.). Die Definitionen und Ausprägungsmöglichkeiten hängen stark vom untersuchten Gegenstand, dem Forschungsanliegen sowie der methodologischen bzw. paradigmatischen Position ab, welche in großer Vielfalt von der Methode Gebrauch machen (Übersicht: Stegbauer/Häußling 2010).

Formal ist ein Netzwerk definiert „als eine abgegrenzte Menge von Knoten oder Elementen und der Menge der zwischen ihnen verlaufenden sogenannten Kanten“ (Jansen 2006: 58). Die Knoten oder Elemente repräsentieren üblicherweise die Akteure[18] und die zwischen ihnen verlaufenden Beziehungen bzw. Relationen werden als Kanten bezeichnet. Letztere können – unabhängig von der numerischen Qualität – als ungerichtete Relationen oder gerichtete Beziehungen, die „Absender“ und „Empfänger“ markieren, vorliegen. Das heißt, während im ersten Fall nur zwischen vorhandener und nicht vorhandener Relation unterschieden werden kann, sind bei gerichteten Beziehungen reziproke, einseitige und gar keine Verbindungen mögliche Konstellationen (Jansen 2006). Zudem lassen sich die Knoten bei unterschiedlicher kategorialer Natur in *n* Gattungen aggregieren und in einem sogenannten *n*-Mode-Netzwerk als verschiedene (disjunkte) Knotensets erfassen, die lediglich über Beziehungen zwischen- und nicht untereinander verfügen können (de Nooy et al. 2011).

Die analytischen Verfahren der Netzwerkanalyse basieren auf mathematischen Graphen, die ein Set an *N* Knoten sowie ein Set der zwischen ihnen definierten Kanten *L* repräsentieren. Diese lassen sich als *n*-x-*n*-Matrix darstellen, wobei die Beziehung *l* zwischen zwei Knoten in der Überschneidung der entsprechenden Zeilen x Spalten eingetragen wird: für eine bestehende Verbindung bei unbewerteten Graphen eine 1 und bei bewerteten Graphen der entsprechende numerische Ausdruck, andernfalls eine 0. Bei gerichteten Beziehungen geben die Zeilen die „ausgehenden“ Verbindungen und die Spalten die „eingehenden“ Verbindungen wieder; die Matrix eines ungerichteten Graphen ist über die Hauptdiagonale entsprechend symmetrisch (Jansen 2006).

[18] Es können auch Objekte oder Ereignisse sein, wie es die Akteur-Netzwerk-Theorie nach Bruno Latour praktiziert (Latour, 2010).



	A	B	C	D	E
A	0	0	0	0	0
B	0	0	0	1	1
C	1	1	0	1	0
D	0	0	0	0	0
E	0	0	0	1	0

Abb. 1: Beispiel eines gerichteten Netzwerkes und dazugehörige Soziomatrix (eigene Darstellung)

Im Folgenden ist zunächst die Begrifflichkeit des Netzwerkes aus einem systemtheoretischen Verständnis heraus zu betrachten, um darauf aufbauend empirische Beobachtungs- und Erkenntnismöglichkeiten in Bezug auf den vorliegenden Gegenstand abzuleiten und einer Netzwerkanalyse zugänglich zu machen.

4.2 Netzwerk und System(theorie)

Das Konzept des Netzwerkes findet sich in der Systemtheorie nach Niklas Luhmann in zweifacher Form: Als Prinzip der (Anschluss-)Operationalität von Kommunikation sowie als Differenzierungsphänomen sozialer Strukturen (Bomme/Tacke 2007). Der erste Fall beschreibt soziale Systeme in Bezug auf das Moment der Autopoiesis als „Systeme, die nicht nur ihre Strukturen, sondern auch die Elemente, aus denen sie bestehen, im Netzwerk eben dieser Elemente selbst erzeugen“ (Luhmann 1997: 65). Der Fokus liegt hier auf der Kommunikation im Sinne eines rekursiven Modus Operandi, der Anschlussmöglichkeiten und deren Realisierung eine netzwerkartige Form attestiert (Luhmann 1997).

> „Damit sind aber nicht die Bedingungen der Verknüpfung zwischen Elementen thematisiert; diese werden systemtheoretisch vielmehr als Konditionierung der Kommunikation durch Strukturen zum Thema, wobei mit Strukturen – anders als in gängigen Netzwerkansätzen – keine Netzwerkstrukturen, sondern stets Systemstrukturen, also kommunikative Sinnstrukturen als Einschränkung von Anschlussmöglichkeiten, gemeint sind“ (Bomme/Tacke 2007: 11f.).

Die zweite Verwendung des Konzepts des Netzwerkes weist im Gegensatz dazu eine deutliche Ausrichtung auf soziale Strukturbildung (von Beziehungen) auf, jedoch ohne konsequenten Anschluss an die Grundbegrifflichkeiten der Systemtheorie (Fuhse 2005). Ausgangspunkt der Überlegungen ist die Freisetzung (polykontexturaler) sozialer Adressen[19] im Zuge funktionaler Differenzierung, die ihre Teilnahme bzw. Inklusion über je eigene (Rollen-)Bedingungen organisieren. Soziale Netzwerke

[19] Der Begriff der sozialen Adresse wird in Anschluss an den Vorschlag von Peter Fuchs (1997) als Konstruktion von Zurechnungspunkten im systemtheoretischen Kommunikationsverständnis mit Bezug auf die Formen gesellschaftlicher Differenzierung verwendet.

– Luhmann hat hier vor allem mafiotische Phänomene vor Augen (Luhmann 1995c) – liegen zu den Funktionssystemen in dem Sinne ‚quer', dass sie über die Verknüpfung von sozialen Adressen unterschiedlicher Verweisungskontexte ‚neue' Möglichkeiten verknüpfen und erzeugen (Bomme/Tacke 2007; Tacke 2000). Die Konstitution von Netzwerken beruht im Gegensatz zu sozialen Systemen auf dem Primat der sozialen Adresse und der darauf aufbauenden Steigerung von Möglichkeiten – systemtheoretisch sind sie dann als Formen sekundärer Ordnungsbildung mit parasitärem Charakter aufzufassen (Tacke 2000).

Ob sich der breite Phänomenbereich sozialer Netzwerke mit dieser Perspektive vollständig erfassen lässt, kann bezweifelt werden, zumal es in Anschluss an die Systemtheorie eine Reihe relativ unterschiedlicher Konzepte in diesem noch recht jungen Forschungsfeld gibt (Fuhse 2005). Als Heuristik lassen sich von dieser Herangehensweise aus jedoch ergiebige Ansatzpunkte zu einer theoretischen Modellierung und empirischen Erfassung des Gegenstandes konzipieren.

Ein soziales Netzwerk, welches sich „über die reflexive Kombination der mit polykontexturalen Adressen verbundenen Möglichkeiten" (Tacke 2000: 293) realisiert, reproduziert und grenzt sich als solches „im Modus einer reziproken Leistungskommunikation [durch] ein mehr oder weniger spezifisches *Leistungsspektrum*" (Tacke 2011: 15) ab.

> „Selbstbezügliche Formen der Einschränkung, d.h. *Grenzziehung*, sind für die Herstellung von sozialen Netzwerken daher notwendig. Die Grenzziehung beruht im Falle von Netzwerken aber offenbar nicht auf einem einzelnen Mechanismus […], sondern auf einer ‚Verschleifung' von sozialen (wer?), sachlichen (was?) und zeitlichen (wann?) Einschränkungen der Netzwerkkommunikation" (Tacke 2011: 16).

Eine solche Konzeption von sozialen Netzwerken eröffnet systemtheoretische Anschlussmöglichkeiten, die über das Ausgangsverständnis sozialer Netzwerke als sekundäre Ordnungsform parasitärer Natur hinausgehen. Mit selektiven Einschränkungen – als Auswahl zugelassener Relationen – im bzw. des sozialen Netzwerkes wird diesem die „Fähigkeit" zur Konstitution von (Erwartungs-)Strukturen und emergenten Sinnordnungen zugeschrieben (Luhmann 1987). Systembildung und (Re-)Produktion ist dann in (Kommunikations-)Netzwerken als solche konzipierbar: als mehr oder weniger ausgeprägte Strukturkondensation und damit verbundene Grenzziehung auf der Basis selektiver, selbstreferenzieller (Kommunikations-)Prozesse unter den Bedingungen doppelter Kontingenz in einem letztlich die gesamte Gesellschaft umfassenden Kommunikationsnetzwerk (Luhmann 1972, 1987). Diese Herangehensweise begreift soziale Netzwerke nicht als eine Sonderform sozialer Systeme, sondern so-

ziale Systeme als spezifische Phänomene bzw. Ausprägungen sozialer (Kommunikations-)Netzwerke.

Anhand dieser Ausgangslage entwickelt Stefan Fuchs (2005) einen Ansatz zur Integration der Systemtheorie in ein netzwerktheoretisches Verständnis sozialer Phänomene. Fuchs geht vom Netzwerk als grundlegendem „Masterkonzept" sozialer Beziehungen aus und konzipiert soziale Systeme als Formen der Grenzziehung und Involution in bzw. durch diese Netzwerkstrukturen. Dabei greift er zentral auf eine weitreichende Begrifflichkeit der Kultur zurück:

> „Assume that a 'culture' is a recursive network of self-observations and –distinctions from other cultures or noncultures. Distinctions create boundaries of varying sharpness and permeability. They produce an inside and outside, separating that which belongs to a culture from that which does not belong" (Fuchs 2005: 156).

Dieses Verständnis von Kultur geht dabei weit über das Luhmanns hinaus, der diesem Begriff einen insgesamt eher unzulänglichen theoretischen Nutzen attestiert (Luhmann 1997). Dennoch rekurriert Luhmann in seinen (späteren) Werken an nicht wenigen Stellen auf diese Begrifflichkeit und eröffnet so eine Reihe verschiedener theoretischer Anknüpfungspunkte. In Bezug auf Organisationen sieht Luhmann Kultur in ihrer Letztkomponente als latenten Wertrahmen, der als anonym hergestelltes Ergebnis der Praxis die unentscheidbaren Entscheidungsprämissen repräsentiert (Luhmann 2000). Nach „innen" liegt ihre Funktion in einer nicht thematisierten Unterstellung von Gemeinsamkeit, die gerade aufgrund ihrer Latenz als Selektionskoordination wirkt, während sie nach „außen" als Abgrenzungsmechanismus fungiert bzw. fungieren kann (Luhmann 2000). In Bezug auf ihre Ordnungsleistung zeigt sie eine enge Verwandtschaft zu Parsons Konzept des „shared symbolic system", jedoch ist für Luhmann Kultur „kein notwendig normativer Sinngehalt, wohl aber eine Sinnfestlegung (Reduktion)" (Luhmann 1987: 224f.).

Damit ist jedoch noch nichts über Konstitution bzw. Prozesslichkeit des Kulturbegriffes gesagt. In einem solchen Zusammenhang bezeichnet Luhmann Kultur als „das Gedächtnis sozialer Systeme […]. Kultur ist, anders gesagt, die Sinnform der Rekursivität sozialer Kommunikation" (Luhmann 1999c: 47). Mittels vereinfachter Zurechnungsprozeduren – grundlegend: Ego/Alter, konstant/variabel, external/internal – schematisiert das Gedächtnis auftretende Ereignisse und konstruiert selektive Bezugspunkte, an die als Ergebnisse im Weiteren angeschlossen werden kann (Luhmann 1996). Operativ „überbrückt" das System damit den temporalen Charakter der Elemente, indem es über kontextübergreifende Schematismen die Elemente abstrahiert und deren vor- und rückgreifende Relationierung ermöglicht (Schmitt 2009). Die Funktion des Gedächtnisses liegt in der Regulation des Vergessens und Erinnerns und (re-)produziert sich als kontinuierliche Selbstbeobachtung der Kommunikation in

jeder Operation des Systems (Luhmann 1996, 1999c). Sich bewährende Schematismen, die über Kondensation und Konfirmation (Redundanz) bzw. Kreuzen und Negieren (Varietät) die strukturbildenden Potenziale der konkreten Ereignisse realisieren, etablieren sich als systemische Eigenwerte in Form von Sinnfiguren, welche die laufenden Kohärenzprüfungen der Kommunikation leiten und den Raum möglicher Anschlusskommunikation selegieren (Luhmann, 1996; Schmitt 2009). Entsprechende Sinnprämissen und Erwartungserwartungen lassen sich dann unter variierendem Grad an Spezifikation und Generalisierung in (selbstbeschreibenden) Semantiken und Mediencodes ausmachen, bei Einschluss von Handlungserwartungen auch als Skripte auffassen (Luhmann, 1996; Schmitt, 2009). Luhmann (1996) spricht in diesem Rahmen auch von der Operationalisierung der Systemidentität.

Das Kulturverständnis von Fuchs lässt sich – wenn auch nicht explizit von ihm hergeleitet – an diesen Punkten anschließen, indem die (Re-)Produktion von Systemen im bzw. als Netzwerk über den Begriff der Kultur als selbstbeobachtende Unterscheidung konzipiert wird: „The inside/outside distinction is a variable accomplishment of a culture as it observes and demarcates itself from other networks“ (Fuchs 2005: 156). Als selbstreferenzieller, d. h. autopoietischer Prozess konstituiert sich die Einheit des Netzwerkes als emergente (Sinn-)Ordnung – hier: Kulturordnung – durch Kommunikationsprozesse, die im Sinne der Gedächtnisfunktion systemspezifische Sinnprämissen und Erwartungsstrukturen konstituieren. Die Funktion des sozialen Gedächtnisses realisiert sich als alle Operationen begleitende Kohärenzprüfung und lässt sich entsprechend über die Kommunikationsdichte zur Identifikation von Systemwelten heranziehen.

> „The inside of the network is to some extent coherent, and it produces and maintains that coherence against entropy and dissolution – until it does, in fact, disintegrate. Coherence is not consensus, but the outcome of sheer connectivity within the network. Networks have higher internal than external connectivity, though this proportion varies also from network to network. Since networks consist of bounded relations, however, they produce some measure of self-similarity; nothing more is implied by ‘coherence’“ (Fuchs 2005: 157).

Die netzwerktheoretische Perspektive setzt in ihrem Verständnis an dyadischen (Kommunikations-)Beziehungen als grundlegende Einheit sozialer Netzwerke an, die als *Relationsgefüge* die Meta-Struktur des sozialen Feldes bilden. Der Fokus liegt auf diesen Relationen und nicht den Knoten als produzierenden bzw. tragenden Einheiten: „Networks consist of relations, not nodes. […] Nodes are outcomes, not sources or origins, of networks“ (Fuchs 2005: 256). Die Knoten werden im Anschluss an die Systemtheorie als Personen oder Rollen begriffen, die durch Adressierung und Attri-

butierung in das Kommunikationsnetzwerk eingebunden werden[20]: „Nodes are selectively activated by their connections, […] the connection constructs a specific *version* of a node, and it does this according to its own, not the node's specifications" (Fuchs 2005: 65). Als psychische Systeme operieren sie geschlossen, sind Umwelt des Netzwerkes und finden als Personen und Rollen im Sinne „individuell attributierte[r] Einschränkung von Verhaltensmöglichkeiten" (Luhmann 1995b: 148) ihre Form der strukturellen Kopplung (Luhmann 1995b).

Im Fokus einer solchen analytischen Perspektive steht das relationale Gefüge des beobachteten Zusammenhanges im Sinne sozialer Wechselwirkungen. Das heißt, das primäre Interesse liegt auf den strukturellen Eigenschaften des Netzwerkes, die in Beziehung zu Leistungen und Funktionen desselbigen gesetzt werden (Hollstein 2006). Angesprochen ist damit eine emergente Ebene sozialer Ordnung und deren (Re-)Produktion über bzw. in spezifischen Konfigurationen, die sich quantitativ erfassen und auswerten lassen. Soziale Systeme konstituieren sich dann als relationales Beziehungsgeflecht im Netzwerk der Kommunikationen, die als Bedingung des Anschlusses strukturierte oder organisierte Komplexität produzieren.

4.3 Interaktion, Gruppe und Organisation

Auf der Grundlage der vorausgehenden theoretischen Überlegungen lassen sich weiterführende netzwerktheoretische Konzeptionen sozialer Systeme anschließen und im Hinblick auf analytische Verfahren ausarbeiten. Das bedeutet, auf die möglichen Ausprägungen von Netzwerkstrukturen zurückzugreifen und diese auf den theoretischen Rahmen der Analyse zu beziehen.

Entsprechend der oben ausgeführten Möglichkeiten der Relationsausprägung innerhalb eines Netzwerkes lassen sich die Beziehungen nach verschiedenen Aspekten – z. B. Relationsanzahl, Reziprozität – und in Bezug auf die Knoten – z. B. auf Äquivalenz – hin überprüfen und als strukturelles Gefüge einer speziellen Qualität darstellen. Diesem strukturellen Variationspotenzial folgend entwickelt Fuchs (2005) ein konzeptionelles Verständnis sozialer Systeme, welches die interne Verbundenheit der Knoten als grundlegendes Kriterium der Systemhaftigkeit einer entsprechenden Kommunikationsmenge ansieht: Ausgeprägte Kommunikationszusammenhänge, die sich als solche von einer Umwelt strukturell abgrenzen lassen, konstituieren – den obigen Ausführungen folgend – operativ eine „eigene" Kultur im Sinne einer systememergenten Realität. Je stärker die strukturellen Schließungen als Verhältnis interner Verbundenheit und externer Abgrenzung, desto ausgeprägter die „Systemkultur" und der jeweilige Systemcharakter. Je nach Größe und Intensität dieser Schließungen

[20] Prinzipiell können nach Fuchs (2005) je nach Analyseansatz verschiedenste Entitäten (Gegenstände, Netzwerke usw.) Knoten eines Netzwerkes sein – überwiegend konzentriert er sich in seinen Ausführungen jedoch auf Personen bzw. Rollen.

lassen sich dabei mit Fuchs Interaktionen, Gruppen oder Organisationen differenzieren (Fuchs 2005).

> „Networks come first. Encounters, groups, and organizations are variable and temporary 'involutions' or condensations of networks. They emerge as certain segments and clusters of a network turn inward, separating themselves to some degree from the overall structure and from the rest of the world. […] By means of distinction, an involution might even acquire an 'identity' for itself, and for other identities. Network involution create or construct their own internal realities" (Fuchs 2005: 191f.)

Eine solche Konzeption sozialer Systeme setzt am operativen Moment des sozialen Gedächtnisses an, welches als alle Operationen – Kommunikationen – begleitende Funktion über Strukturaufbau die (Re-)Produktion des Systems ermöglicht und leitet. Fuchs' (2005) Klassifikation sozialer Systeme leitet sich aus diesem Verständnis ab und baut auf dem Aspekt der Rekursivität sozialer Kommunikation auf:

Ausgangpunkt der Systemklassifikation ist die Interaktion, die übereinstimmend mit Luhmann (1972) als Kommunikation unter Anwesenden definiert wird und sich über das Kriterium der Wahrnehmung konstituiert und abgegrenzt. Für den vorliegenden Zusammenhang wird diese Konstellation ausschließlich als Kommunikationseinheit, die von der Dimension der Wahrnehmung absieht, interessant – ähnlich der Kommunikation zweier Akteure mittels Brief. Nehmen Interaktionen zwischen mehreren, überwiegend denselben Akteuren jedoch regelmäßigen Charakter an, sieht Fuchs (2005) die Möglichkeit zur Konstitution eines zeitlich stabilen Systemtyps eigener Qualität – der Gruppe. Als Ergebnis der Kommunikationsprozesse unter diesen Bedingungen verweist Fuchs auf die Kondensation systemimmanenter Strukturwerte, welche die Gruppe als System von einer Umwelt abgrenzen und über Anschlussmöglichkeiten eigener Qualität reproduzieren. Als Systemidentität umfasst dies mit Fuchs (2005) einen (stabilen) emergenten Sinn- und Referenzhorizont, der sich im selektiven Modus des Operierens der Gruppe realisiert und insbesondere in der sozialen Dimension in Personen- und Rollenkonstrukten generalisiert. Der Gruppe wird eine maximal angehbare Größe von ungefähr zehn „Mitgliedern" zugeschrieben, die eine enge Kopplung zwischen ihnen erlaubt und aufweist. Ausgeprägte Kommunikationsrelationen werden aus dieser Perspektive als „In-Position-Halten" der Personen angesehen (Einschränkung von Freiheitsgraden), die sich als redundantes Moment in Routinen und Institutionalisierungen ausprägen und als grenzziehende Sinnprämissen auch der graduellen strukturellen Schließung der Gruppe bedürfen (Fuchs 2005). Sehr ähnliche Vorstellungen finden sich bei Luhmann (1999a) bezüglich seiner Ausführungen zur Clique, die jedoch ausschließlich auf formale organisationale Zusammenhängen zielen. Organisationen umfassen aus dieser Perspektive eine Vielfalt möglicher Strukturausprägungen, die sowohl Interaktionen als auch Grup-

penbildung enthalten und sich an netzwerktheoretische Strukturkonzeptionen von Hierarchie und Macht anschließen lassen (Fuchs 2005)[21].

Auch hier bleibt das grundlegende Verbindungsglied zwischen Netzwerk- und Systemtheorie der systemkonstituierende Strukturaufbau auf der Grundlage der alle Kommunikationen begleitenden Gedächtnisfunktion sozialer Systeme. An diesem Punkt der theoretischen Konzeption zeigt sich jedoch auch die grundlegende analytische Bedeutung des kontextuellen Rahmens und die methodischen Grenzen eines netzwerktheoretischen Zugangs: Strukturaufbau vollzieht sich auch – das betrifft vor allem Organisationen, aber auch Interaktionen – unter prägenden „Vorgaben" einer (teilsystemspezifischen bzw. inneren) Umwelt (Luhmann 1997; Schmidt 2007). Diese „Vorgaben" realisieren sich zwar unter der Bedingung der Autopoiesis durch die Operationen des jeweiligen sozialen Systems selbst, jedoch lassen sich mit zunehmender (System-)Differenzierung auch verschiedene Kommunikations- und Strukturebenen unterscheiden, die in einer strukturellen Netzwerkanalyse Beachtung finden müssen, will man soziale Systeme abgrenzen[22].

Der vorliegende Gegenstand zeichnet sich unter diesem Aspekt durch die Möglichkeit eines relativ pragmatischen Zugangs zur Kommunikationsqualität aus: Die internetbasierte Kommunikation im Rahmen des Softwareprojekts wird zentral über die verschiedenen Projekt- bzw. Organisationstools in Form virtueller Räumlichkeiten vorstrukturiert, die als Medium der Kommunikation durch Aufbau und Differenzierung den (latenten) Sinnhorizont des betrachteten Zusammenhanges tragen und repräsentieren (s. Kap. 3). Auf dieser Grundlage lässt sich die strukturelle Analyse des Kommunikationsnetzwerkes in Beziehung setzen zu den die Kommunikation leitenden Schemata und Relevanzstrukturen. Relationale Kommunikationsgefüge können vor diesem Hintergrund einer Interpretation ihrer Funktion und Leistung im organisationalen Kontext der Softwareproduktion unterzogen und einer abschließenden, umfassenden Perspektive zugeführt werden.

---

[21] Die entsprechenden Konzepte knüpfen an strukturanalytische Verfahren der Netzwerktheorie an, die hier nicht im Einzelnen ausgeführt werden können, aber der entsprechenden Basisliteratur entnehmbar sind (Jansen 2006; Trappmann et al. 2011). Analyseverfahren, die im Rahmen dieser Arbeit Anwendung finden, werden an entsprechender Stelle beschrieben und die zentralen theoretischen Anschlussmöglichkeiten erläutert.

[22] Fuchs (2005) führt diesen Aspekt nicht explizit aus, jedoch ist anhand seiner Ausführungen zu erkennen, dass der alleinige (quantitative) Bezug auf das Netz der Kommunikationen dem systemtheoretischen Rahmen nicht gerecht würde. So fordert Fuchs, in der netzwerkbasierten Erfassung von Organisationen zwischen formeller und informeller Kommunikation zu unterscheiden, was eine entsprechende Konzeption dieses Unterschieds sowie die Erfassung der Kommunikations*qualität* voraussetzt. Damit deuten sich die Grenzen eines (ausschließlich) netzwerktheoretischen Ansatzes an, die letztlich in den Beschränkungen im Zugang zur Dimension des Sinns gründen und auf die Bedeutung dieser Dimension im Rahmen sozialer Zusammenhänge verweisen. Am deutlichsten zeigt sich diese Problematik vielleicht gerade in dem Umstand, dass Fuchs seine netzwerktheoretische Systemkonzeption mit dem Systemtyp Organisation beendet und nicht auf der Ebene der Funktionssysteme fortführt, wo dieser Aspekt (Stichwort: symbolisch generalisierte Kommunikationsmedien) nochmals gesteigerten Nachdruck erlangen würde.

5 Das Softwareprojekt „Password Safe“

5.1 Datengrundlage

Die Betreiber der Plattform SourceForge.net stellen seit dem Jahr 2003 (ab Feb. 2005 monatlich) in Zusammenarbeit mit der University of Notre Dame eine Sammlung der wesentlichen Informationen über die Plattform, seine Nutzer und deren Aktivitäten in Form einer relationalen Datenbank zu Forschungszwecken zur Verfügung (Madey). Nach Beantragung einer Zugangsberechtigung kann die Datenbank mittels SQL (Structured Query Language) über ein Abfrageformular auf einer Webseite[23] der University of Notre Dame angesteuert werden, die gesuchten Informationen können abgefragt und in einer Texteditor-Datei gespeichert werden. Über eine eigene Wiki-Homepage erhält man Zugang zu Informationen zur Datenbank und deren Struktur (Entity-Relationship-Diagramme) sowie bestehende Forschungsergebnisse und -artikel (SourceForge Research Data Archiv).

Die Auswahl eines Softwareprojekts aus mehr als 300.000 Projekten ist kein einfaches Unterfangen, da sie aufgrund der großen Grundgesamtheit einen recht restriktiven Charakter aufweisen muss und die Reichweite und Geltung der Untersuchungsergebnisse eines exemplarischen Projekts grundlegend bestimmt. Dieser Umstand muss sowohl im Auswahlprozess als auch insbesondere in der abschließenden Bewertung der Ergebnisse berücksichtigt werden. Die Auswahlkriterien im vorliegenden Fall orientieren sich an zwei Momenten: (1) der Beschaffenheit der Plattform und (2) der gewählten Forschungsthematik:

SourceForge.net bietet seinen Nutzern wie oben ausgeführt eine Reihe von Tools[24] zur Koordination und Unterstützung des Softwareentwicklungsprozesses. Dabei handelt es sich zunächst einmal um ein allgemeines Angebot, aus dem die Entwickler wählen können. Die Nutzung ist keinesfalls obligatorisch und erfolgt nicht selten recht selektiv – entsprechende Dienste bzw. Leistungen werden dann in der Regel über externe Homepages und Programme realisiert (Howison/Crowston 2004). Um an eine möglichst vollständige Netzwerkstruktur aus den Dokumentationsdaten der Plattform zu gelangen, sollten die zur Auswahl stehenden Projekte daher ihre Tätigkeiten möglichst ausschließlich über die Plattform koordinieren bzw. vollziehen.

Die Forschungsthematik der vorliegenden Arbeit konzentriert sich auf die organisationale Strukturierung – hier: der Kommunikationsstrukturen – von Projekten freier/offener Softwareentwicklung im Internet. Um diesen Gegenstand nachhaltig zu untersuchen, sollte das ausgewählte Projekt daher zum einen möglichst aktiv sein, um ei-

[23] http://srda.cse.nd.edu/cgi-bin/form.pl (27.08.2012).
[24] Im Weiteren werden zur Bezeichnung dieser Tools (Tracker, Foren etc.) die Begriffe Organisationstools, Kommunikationstools oder nur Tools synonym verwendet.

ne solide Datengrundlage zu gewährleisten, und zum anderen über „gefestigte“ Strukturen verfügen, d. h. ein gewisses Alter und einen gewissen Erfolg aufweisen.

5.2 Projektauswahl

Im vorliegenden Auswahlverfahren wurde anhand einer Reihe quantitativer Vorgaben versucht, die obigen Vorgaben zu sichern. Die Abfragesyntax (SQL) der einzelnen Schritte befindet sich im Anhang (vgl. Anhang II). Die abgefragten Daten wurden mit Excel eingelesen und verarbeitet. Die berechneten statistischen Kennzahlen dienen ausschließlich der Sicherung der Zielvorgaben im Auswahlverfahren und haben einen approximativen Charakter.

Selektion I

Aus der Gesamtheit aller auf der Plattform verzeichneten Projekte wurden in einem ersten Schritt jene selektiert, die Mail-, Forums-, Projektmanagement-, News-, SVN- und Beschreibungsfunktion aktiviert haben, sich auf der Entwicklungsstufe „Produktion/Stable“ oder „Mature“ befinden, mindestens ein Jahr (365 Tage) alt sind und in der Zeit bis zum Erhebungsstichtag (15. Mai 2011) mindestens ein Artifact, eine Message und eine frs-File produziert haben[25].

Die vier Kriterien sollen sicherstellen, dass nur jene Projekte einbezogen werden, welche die grundlegenden Koordinationstools der Plattform annehmen (i) und tatsächlich nutzen (iv) sowie auf einer elaborierten Entwicklungsstufe (ii) angelangt sind, die eine erfolgreiche Systembildung nahelegt. Das Mindestalter von 365 Tagen (iii) dient ebenso der Sicherung einer nachhaltigen Entwicklungsgeschichte wie dem Ausschluss von Projekten, die sich möglicherweise erst in einer späten Phase auf der Plattform anmelden. Die Anzahl der Projekte, die den obigen Vorgaben genügen, ist mit N = 2.297 stark reduziert worden.

Selektion II

In einem zweiten Schritt wurde für die 2.297 Projekte die Gesamtanzahl aller produzierten Artifacts, Messages und frs-Files abgefragt und in Relation zum jeweiligen Alter in Tagen gesetzt. Anschließend wurde für jeden der drei Werte der Quantilsrang berechnet und all jene Projekte ausgeschlossen, bei denen einer der Werte unter 0,9 lag. Die Selektion mittels dieses Lagemaßes sollte die im Vergleich aktivsten Projekte herausfiltern und sicherstellen, dass nicht nur ausgewählte Funktionen der Plattform (z. B. nur die Bereitstellung der aktuellen Softwareversion über das File Management System) genutzt werden. Zudem wurden die gewählten Aktivitätsmaße in der

[25] Artifact bezeichnet Tickets bzw. Anliegen in den Fallbearbeitungssystemen (Issue Tracking System) der Projekte, Message Nachrichten in den Foren der Projekte und frs-File eine im File Release System hinzugefügte bzw. freigegebene Datei (s. o.). Die Einstufung des Reifestatus wird seitens der Plattform vorgenommen.

bestehenden Forschung zum Open-Source-Bereich bereits als Erfolgs- bzw. Gütemaß der Softwareentwicklung beschrieben und werden als Voraussetzung für belastbare Netzwerkanalysen angesehen (Crowston et al. 2006). Die Selektion nach obigen Kriterien verringert die Auswahl auf 45 Projekte.

Selektion III

Im dritten Schritt des Auswahlverfahrens wurde zentral auf die (ausgeglichenen) Projektaktivitäten innerhalb des angestrebten Erhebungszeitraumes (15. Februar 2011 bis 15. Mai 2011) abgezielt. Diese wurden wie im vorherigen Fall über die Anzahl der produzierten Artifacts (‚Eröffnet' oder ‚Geschlossen'[26]) und Messages sowie, anstelle der frs-Files, über die Reading- und Writing-Aktivitäten im Source Code Management System (SCM) erfasst[27]. Die späte Umstellung auf die genaueren SCM-Kennwerte ist den Schwierigkeiten geschuldet, diese Werte über den SourceForge.net-Datensatz abzufragen, so dass sie direkt über die Projektstatistik auf der jeweiligen Projektseite erhoben werden müssen. Ein solches Vorgehen ist jedoch erst bei kleineren Fallzahlen realisierbar.

Ausgeschlossen wurden die Projekte, deren Kennwerte für den Erhebungszeitraum nicht jeweils eine Ausprägung ≠ 0 aufweisen. Die verbleibenden 13 Projekte wurden auf der Basis der Ausprägungen ihrer Aktivitätsmerkmale ‚Artifacts', ‚Messages' und ‚SCM' in eine Rangfolge[28] gebracht, welche die abschließende Grundlage für die Projektauswahl darstellt. Die ersten beiden Projekte – „Adempiere ERP Business Suite" und „Password Safe" – weisen als einzige der verbleibenden Projekte in jeder der drei Variablen eine Ausprägung auf, die über dem 0,75-Quantil liegt. Das heißt, ihre Aktivitätskennwerte gehören in jedem der drei Merkmale zu den höchsten 25 % aller Beobachtungswerte in dieser Vergleichsgruppe.

[26] Gezählt werden alle Artifacts, die in diesem Zeitraum eröffnet oder geschlossen werden. Es kann also zu doppelten Erfassungen kommen, wenn das Anliegen in diesem Zeitraum sowohl angelegt als auch abschließend bearbeitet wurde. Als Kennzahl für die Aktivität des Projekts ist dies jedoch relativ unerheblich, da beide Akte (Eröffnen und Schließen) auf Tätigkeiten innerhalb des Projekts verweisen.

[27] SCM-Kennzahlen bieten den Vorteil, dass sie explizit die Aktivitäten am Quellcode wiedergeben und somit deutlich aussagekräftiger sind als die Kennzahlen des Files Release System (frs), welche nur über die Veröffentlichung neuer Software(versionen) Auskunft geben. Zwischen solchen Veröffentlichungen können größere Zeitabstände auftreten, die jedoch keinen direkten Rückschluss auf die Programmieraktivitäten zulassen. Für einen längeren Zeitraum kann man, wie in den vorherigen Selektionsschritten, den Aktivitäten im File Release System einen approximativen Informationswert beimessen. Für kleinere Zeiträume, wie den hier vorliegenden dreimonatigen Erhebungszeitraum, eignen sich die frs-Kennwerte für Aussagen über die Programmieraktivitäten nicht. Da die vorhandenen SCM-Kennwerte jedoch Schreibe- und Leseaktivitäten zusammenfassen, müssen auch sie mit Vorsicht behandelt werden.

[28] Für die Projekte wurde für jede der drei Variablen Artifacts, Messages und SCM eine Rangfolge (größter Wert = Rang 1) erstellt. Die Ränge wurden im Folgenden für jedes Projekt in einer neuen Variablen summiert, welche in eine abschließende Rangfolge (größter Wert = Rang 1) überführt wurde.

Name	Entwicklungs-stufe	Arti-facts	Messa-ges	SCM[29]	Down-loads	Ra-ting	Bewertun-gen
ADempiere ERP Busi-ness Suite	Production/ Stable	99	1284	42041 (582)	43279	90 %	253
Password Safe	Production/ Stable	50	115	188081 (193)	165023	89 %	795

Tab. 1: Aktivitätskennzahlen der zwei hochrangigsten Projekte im Erhebungszeitraum (15. Februar 2011 bis 15. Mai 2011)

Beide Projekte erfüllen die Vorgaben, die zu Beginn des Auswahlverfahrens festgelegt wurden: Es handelt sich um aktive und erfolgreiche Projekte, welche ihre Softwareentwicklung über die SourceForge.net-Plattform koordinieren, sich auf einer stabilen Entwicklungsstufe befinden sowie hohe Downloadzahlen und gute Bewertungen aufweisen. Aufgrund des mit etwa elf Jahren deutlich längeren Bestehens des Projekts „Password Safe“ im Vergleich zu den ungefähr fünf Jahren des Projekts „ADempiere ERP Business Suite“ und vor dem Hintergrund, eine Auswahl treffen zu müssen, die sowohl eine belastbare Datengrundlage als auch ein in diesem Rahmen handhabbares Datenaufkommen sichert, fiel die Entscheidung letztlich zugunsten des Projekts „Password Safe“.

5.3 Netzwerkerhebung

Das Projekt „Password Safe“ entwickelt eine Software in der Programmiersprache C++ für die Betriebssysteme Windows und Linux, welches die Speicherung und Sicherung einer Mehrzahl von Passwörtern mittels eines Passwortes seitens des Nutzers ermöglicht. Es richtet sich primär an Endnutzer und Systemadministratoren, soll aber auch andere Entwickler ansprechen und ist in neun Sprachen verfügbar (PasswordSafe Projektplattform). Das Projekt besteht seit dem 29. November 2001 und listet zum Stichtag des 15. Mai im vorliegenden Datensatz der Plattform insgesamt 43 Projektmitglieder.

Grundlage des Netzwerkes sind alle Kommunikationsakte, welche im Zeitraum zwischen dem 15. Februar 2011 und dem 15. Mai 2011 innerhalb des Projekts über die Plattform getätigt wurden (vgl. Anhang III). Dazu zählen:

i) Beiträge (Messages) in den Foren „Help“ und „Open Discussion“,

ii) Beiträge (Messages) in den Fallbearbeitungssystemen „Bugs“ (Fehler), „Feature Request“ (Leistungsanforderung) und „Support Request“ (Unterstützungsanforderung),

[29] In Klammern: die absolute Anzahl der vermerkten Einträge (Revisionen) im Protokoll (Log) der jeweiligen SCM-Tools.

iii) Beiträge im Newsboard des Projekts und

iv) Quellcodebeiträge in Source-Code-Management-Tool „Subversion“.

Alle registrierten User der Plattform erhalten eine individuelle ID, welche die eindeutige Zuordnung der einzelnen Kommunikationsakte zum Verfasser bzw. Absender erlaubt. Nicht mehr registrierte Nutzern wird einheitlich die ID 100 zugewiesen, was gegebenenfalls ein nicht unerhebliches empirisches Problem darstellen kann (Howison/Crowston 2004). Im vorliegenden Fall stammen von insgesamt 523 Kommunikationsakten jedoch nur neun Fälle von solchen Nutzern, die aufgrund ihrer geringen Anzahl aus der Analyse ausgeschlossen wurden. Die übrigen Fälle wurden nach dem folgenden Schema angeordnet, das die Grundlage für die Erstellung eines 2-Mode-Netzwerkes bildet:

User ID	**Event**	**Anzahl**
190192	Bug15	7
226518	CVS	17
370700	News	1
370700	Discussion13	1
N = 65	**N = 94**	**N = 480**

Tab. 2: Auszug aus der Datenmatrix zur Erstellung des Netzwerkes

Die erste Spalte identifiziert mittels der User ID den Absender der Kommunikation, die in der zweiten Spalte dem jeweiligen Event innerhalb der Organisationstools zugeordnet wird, während die dritte Spalte die Anzahl der getätigten Mitteilungen zusammenfasst. Um den Informationswert der zur Verfügung stehenden Daten möglichst vollständig zu erhalten, wurde auch die bestehende Differenzierung *innerhalb* der Organisationstools übernommen und kodiert. Das heißt: Im Fallbearbeitungssystem wurde unterschieden zwischen den Untergruppen „Bugs“, „Feature Request“ und „Support Request“, bei den Foren zwischen „Help“ und „Open Discussion“ und bei den Mailing Lists zwischen „Announce“ und „Linux“ (2. Ebene). Innerhalb dieser „thematischen“ Teileinheiten lassen sich dann die einzelnen Anliegen – Tickets in den Fallbearbeitungssystemen und Threads in den Foren – einzeln erfassen und den Mitteilenden zuordnen (3. Ebene). Der alphabetische Teil der Kodierung (z. B. „Bug“) verweist auf die Unterteilung der Organisationstools, während der anschließende numerische Ausdruck die einzelnen Anliegen innerhalb dieser Tools codiert und identifiziert (Abb. 2).

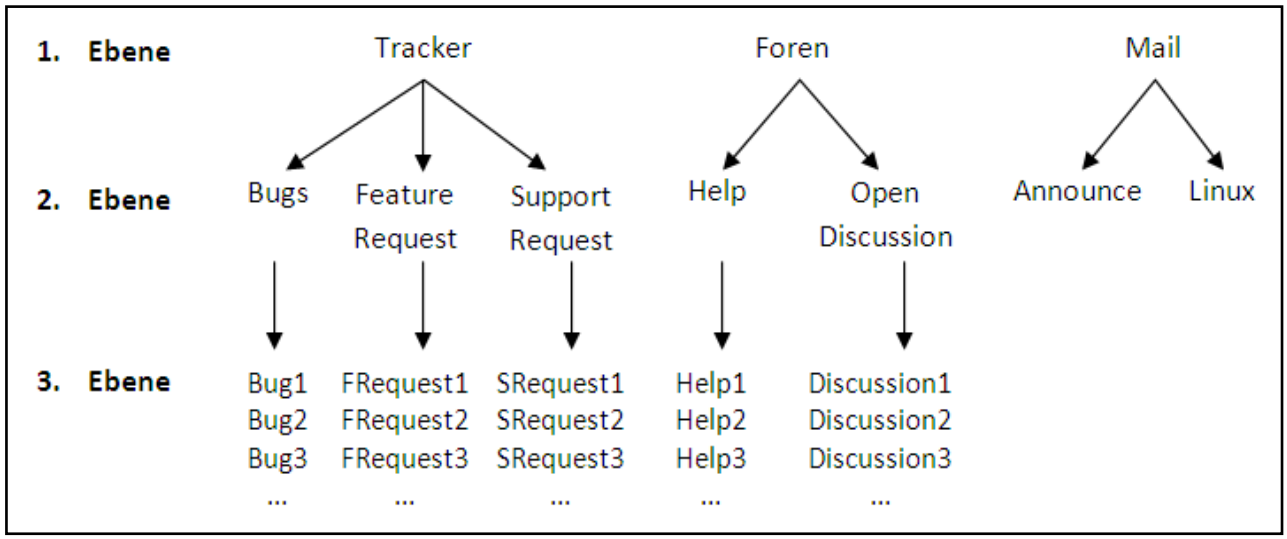


Abb. 2: Differenzierung der Organisationstools Tracker, Forum und Mail (eigene Darstellung)

Die Datentabelle wurde in eine Netzwerkmatrix für die Analyseprogramme Pajek (Mrvar/Batagelj) und UCINET (Borgatti et al. 2002) transformiert, die als 2-Mode-Version sowohl die aktiven Nutzer der Plattform als auch die Events der verschiedenen Organisationstools (entsprechend der Ebene 3) enthält. Es ergeben sich insgesamt 159 Knoten, die sich in 65 Akteure und 94 Events unterteilen. Diese sind verbunden über insgesamt 187 Kanten, deren jeweiliger Wert die Anzahl der getätigten Mitteilungen der Akteure zum betreffenden Event ausdrückt.

Die im Erhebungszeitraum erfassten Plattformnutzer unterteilen sich formal in 6 (7)[30] Entwickler – 5 Developer und 1 Administrator[31] – und 58 User, die insgesamt 480 Mitteilungen auf der Projektseite abgelegt haben. Diese verteilen sich auf 10 im Erhebungszeitraum genutzte Kommunikationsstellen: Quellcodemanagement „CVS“ (1), Forum „Open Discussion“ (2), Forum „Help“ (3), „Bug“-Tracker (4), „Feature Request“-Tracker (5), „Support Request“-Tracker (6), Mailinglist „Linux“ (7), Mailinglist „Announce“ (9) sowie das Newsboard „News“ (10). Abbildung 3 zeigt die Häufigkeitsverteilung der anfallenden Events – nicht der Kommunikationen![32] – innerhalb der Organisationstools, welche einen Überblick über die im Erhebungszeitraum anfallenden Aktivitäten und deren verhältnismäßige Verteilung geben.

[30] Ein projektfremder Entwickler (Developer6) beteiligt sich im Erhebungszeitraum im Rahmen einer Diskussion zu einer Übertragung der Software auf mobile Hardware (s. u.).
[31] Im Weiteren immer als „Admin“ bezeichnet, der jedoch der Gruppe der Entwickler zuzurechnen ist.
[32] Im Falle des CVS sind Events und Kommunikationen äquivalent, da hier jeweils eine Einheit (Revision) geschlossen im System abgelegt wird und einsehbar ist.

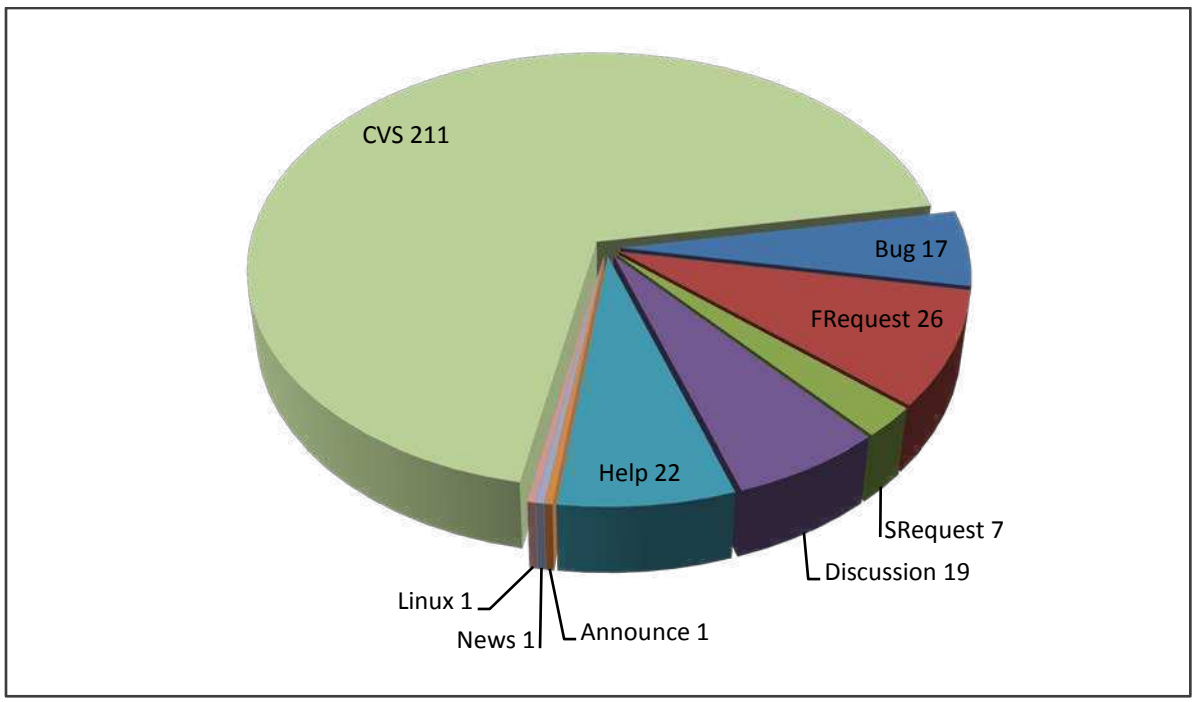


Abb. 3: Häufigkeitsverteilung der Events nach Organisationstools

Der Großteil der Aktivitäten im Projekt entfällt auf die Programmierung der Software (CVS), welche die primäre Kernaktivität des vorliegenden Arbeitszusammenhanges darstellt. Von öffentlichen Mitteilungen über Mailverteiler (Linux, Announce) und Newsboard (News) wird relativ selten Gebrauch gemacht, während die Tracker- und Forenaktivitäten etwa ein Viertel der gesamten Projektaktivitäten ausmachen. In diesem Bereich, an dem Entwickler *und* User beteiligt sind, zeigt sich zudem das inhaltlich-strukturelle Kooperationsaufkommen offener Softwareentwicklung am deutlichsten: Der Trackerbereich (Bug, Frequest, SRequest), welcher sehr konkrete Anliegen in Bezug auf die Software und den Quellcode formuliert, weist mit insgesamt 51 Tickets ein recht ausgewogenes Verhältnis zu den 41 Threads des Forenbereiches (Help, Discussion) auf. Letztere umfassen sowohl allgemeine Ansuchen um Hilfe, meist bei Problemen in der Anwendung der Software, und bieten Raum für thematisch offene Diskussionen ohne direkt vorgegebenen Bezug auf die Software – beides wiederum in einem relativ ausgewogenen Verhältnis. Tracker- und Forentools, die sowohl inhaltlich als auch intentional im Sinne einer inhärenten Gebrauchslogik differieren (s. o.), werden im Kooperationszusammenhang des Softwareprojekts relativ proportional in Anspruch genommen[33].

[33] Dieses Ergebnis ist in gewissem Maße auf einen Bias der Projektauswahl zurückzuführen, da die herangezogenen Selektionskriterien auf eine möglichst hohe Ausprägung *aller jeweiligen* Aktivitätskennzahlen zielten. Letztere repräsentieren jedoch vereinfachte, approximierte Kennzahlen, welche die Aktivitäten des Projekts nicht vollständig erfassen; dies wurde erst bei der Netzwerkerhebung des Projekts nach der Auswahl geleistet. Entsprechend können die Kennzahlen des Auswahlverfahrens und die aggregierten Maßzahlen des Netzwerkes nur bedingt verglichen werden, und die obige Dar-

5.4 Netzwerkanalyse

5.4.1 Komponenten

Eine grundlegende Einsicht in die Zusammensetzung und den Aufbau des Netzwerkes lässt sich in einem ersten Schritt über die visuelle Darstellung des Gesamtnetzwerkes gewinnen. Abbildung 4 bildet alle im Erhebungszeitraum anfallenden Kommunikationen in Beziehung zu den einzelnen Events und Akteuren in Form eines 2-Mode-Netzwerkes ab. Die kreisförmigen Knoten repräsentieren dabei die Akteure – rot für eingetragene Entwickler und blau für Anwender –, während die Quadrate die innerhalb der Organisationstools anfallenden Events (Anliegen) wiedergeben; deren Farbgebung folgt der Unterteilung der Tools, wie sie auf der Projektplattform angelegt ist (s. a. Tab. 2: Ebene 2). Die Kodierung der Knoten erfolgt nach einem äquivalenten Schema: Bei den Akteuren lassen sich Anwender und Entwickler unterscheiden, die zum Zwecke der Identifikation fortlaufend nummeriert wurden. Bei den Events bestimmt der erste, alphabetische Teil der Kodierung das betreffende Organisationstool, während der nummerische Ausdruck das betreffende Event festhält (Abb. 4).

Im Netzwerk lassen sich eine große Hauptkomponente und insgesamt 11 Subkomponenten, welche keine Verbindung zu diesem Teil aufweisen, identifizieren; letztere wurden in der Abbildung mittels der nummerierten Mengenkreise hervorgehoben (Abb. 4). Bei der Isolierung der Komponenten könnte es sich jedoch aufgrund des begrenzten Erhebungszeitraumes auch um statistische Artefakte handeln. Um ihre Bedeutung richtig einzuschätzen, insbesondere da sie im Rahmen späterer Analysen infolge methodischer Implikationen ausgeschlossen werden, wurden die betreffenden Fälle gezielt über die Datenbank sowie die ausführliche Dokumentation der Projektseite angesteuert und in ihrer zeitunabhängigen Gesamtheit betrachtet.

In fünf Fällen (Komponenten 1–5) handelt es sich um isolierte Einzelkomponenten, d. h., die Mitteilungen der Akteure finden keinen Anschluss. Inhaltlich scheint dabei eine der fünf Mitteilungen im Hinblick auf ein projektfremdes Forum verfasst worden zu sein, bei einem weiteren Eintrag handelt es sich um ein versehentliches Duplikat, während die anderen drei Beiträge keine auffälligen Abweichungen aufweisen[34]. Die übrigen Subkomponenten repräsentieren allesamt Kommunikationszusammenhänge, die sich über eine Mehrzahl von anschließenden Kommunikationseinheiten mit schwankender Anzahl an ‚tragenden' Personen konstituieren, jedoch mittels des Erhebungszeitraumes nur ausschnittsweise erfasst wurden. Die Komponenten 6, 7 und

stellung dient in diesem Sinne der exakten Angabe der strukturellen Aktivitätsausprägungen des hier behandelten Netzwerkes.

[34] Die Einschätzung dieser Fälle muss sich auf einer spekulativen Grundlage bewegen: Im Allgemeinen ist es in dem Projekt üblich, unabhängig vom Kommunikations*erfolg* eine Rückmeldung zu geben. Da die Wahrscheinlichkeit des Erreichens sehr hoch ist, ist als ‚Ursache' für das Unterbleiben einer Anschlusskommunikation ein kritischer Informationswert zu vermuten (Luhmann 1981b).

8 sind in ihrem Gesamtumfang relativ kurze Kommunikationsepisoden mit alleiniger Beteiligung von Usern, die auch zeitunabhängig nicht in das Gesamtnetzwerkes eingebunden werden. Es handelt sich bei diesen Fällen ausschließlich um Hilfsanfragen: In zwei Fällen (7, 8) scheint das Anliegen unter den Usern geklärt worden zu sein; im anderen Fall (6) teilten mehrere User das gleiche Anliegen, welches jedoch nicht beantwortet wurde. Die Komponenten 9 und 10 wären bei zeitlicher Ausdehnung des Erhebungszeitraumes eingebunden in die Hauptkomponente des Netzwerkes, da in ihrem zeitlichen Verlauf auch Entwickler an den Diskussionen teilnehmen. Komponente 11 hingegen stellt ein thematisch geschlossenes Subnetzwerk mit einer Vielzahl an Diskussions- und Hilfsthreads dar, an dem sich eine relativ große Anzahl an verschiedenen Akteuren beteiligte und welches zur Gründung eines entsprechenden Schwesterprojekts („pwdsafe-iphone“) führte. Es handelt sich in diesem Fall um eine Subkomponente des Netzwerkes, die keinen direkten Anschluss an die Hauptkomponente findet und sich im weiteren Verlauf auf einer eigenen Projektseite organisiert; Developer6 ist dabei eingetragener Entwickler dieses neuen, eigenständigen Projekts.

Überprüft man auch die Events der Hauptkomponente des Netzwerkes, die über lediglich einen In-Degree verfügen – in Abb. 4: die gestrichelten Mengenkreise – und somit potenziell „erfolglose“ Kommunikationsangebote darstellen[35], auf Anschlusskommunikation über den gewählten Erhebungszeitraum hinaus, zeigt sich hier nur ein Fall (Help19), der ohne Anschluss bleibt. Alle anderen Mitteilungen sind in eine mindestens zwei Akteure umfassende Kommunikationskonstellation eingebunden.

Insgesamt lässt sich festhalten, dass der Großteil der Subkomponenten auch bei zeitlicher Ausdehnung des Erhebungszeitraumes seinen Isolationsstatus nicht verlieren würde. Hervorzuheben ist in diesem Zusammenhang, dass es sich bei den betreffenden Fällen mit einer Ausnahme ausschließlich um Foren handelt und die Beteiligung der Entwickler im Vergleich zur Hauptkomponente des Netzwerkes relativ gering ist. In Zusammenschau mit der inhaltlichen Überprüfung der entsprechenden Fälle zeigt sich, dass Hilfsanliegen oder Diskussionen durchaus ohne Entwicklerbeteiligung auskommen und sich thematisch zwar verwandte, aber dennoch die hier entwickelte Software nicht direkt betreffende Kommunikationszusammenhänge über die Projektseite ausbilden und reproduzieren. Die Anzahl der Mitteilungen, die keinen Anschluss finden, ist dabei sehr gering: Bei über den Erhebungszeitraum hinausgehender Gesamtbetrachtung erfolgt nur in 6 von 92 Fällen keine Anschlusskommunikation.

[35] Die Events „Announce“ und „News“ müssen aus dieser Perspektive explizit ausgeschlossen werden, da es sich um 1-Weg-Kanäle handelt. Die betroffenen Fälle wurden in Abb. 4 durch die gestrichelten Linien zusammengefasst und markiert.

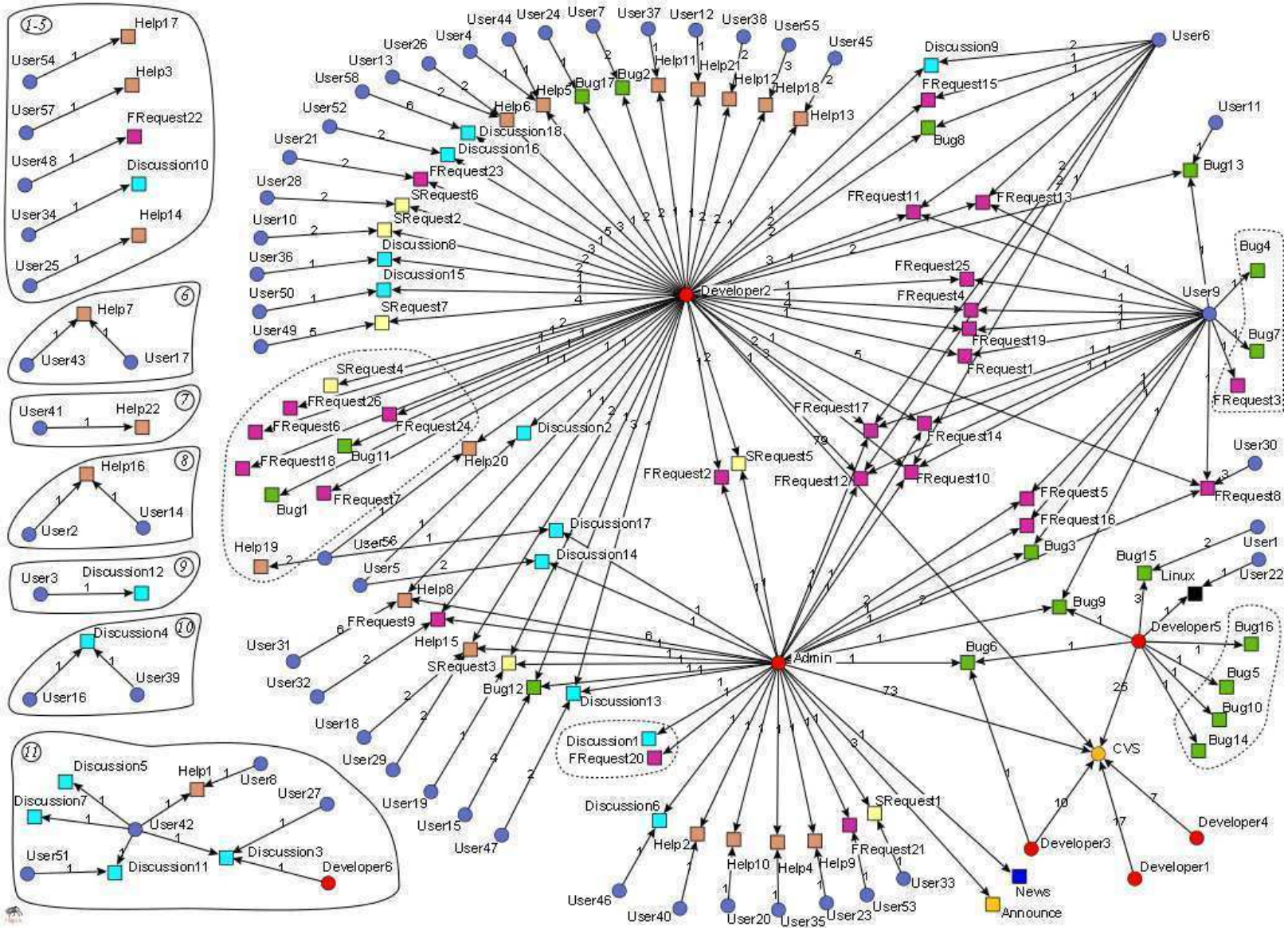


Abbildung 4: 2-Mode-Darstellung des Gesamtnetzwerkes

5.4.2 Allgemeine Kennzahlen

Um einen übersichtlichen Einblick in die Kommunikationsstrukturen des Projekts zu gewinnen, empfiehlt es sich, die einzelnen Events nach ihrer Zugehörigkeit zu den verschiedenen Kommunikationstools und deren thematischer Untergliederung (2. Ebene) zusammenzufassen (Abb. 5). Auf diesem Wege kann man die organisationale Differenzierung der Kommunikationsstrukturen abbilden und grundlegende Erkenntnisse über die Organisationspraxis des Projekts gewinnen. Über die Degree-Verteilungen des Ausgangsnetzwerkes und der unten abgebildeten aggregierten Variante lassen sich ergänzend eine Reihe statistischer Kennzahlen gewinnen, die in Beziehung zur visuellen Analyse des Netzwerkes gesetzt werden können (Anhang IV). Um eine der „organisationalen Wirklichkeit" möglichst entsprechende Datengrundlage zu haben, wurden gemäß der obigen Ausführungen alle Subkomponenten des Netzwerkes und die Events mit lediglich einem In-Degree von der Analyse ausgeschlossen. Ebenfalls nicht berücksichtigt wurden die Organisationeinheiten „News" und „Announce" aufgrund ihres speziellen Mediencharakters (1-Weg-Kanal)[36].

Die Darstellung des Netzwerkes in Abbildung 5 zeigt die Kommunikationen und ihre Absender in Zuordnung zu den virtuellen Räumlichkeiten der Projektseite als gewichtetes 2-Mode-Network. Die Kantenwerte geben die Anzahl der Mitteilungen wieder, welche die Akteure zu den jeweiligen Organisationstools beisteuern. Die Anzahl der innerhalb der jeweiligen Organisationstools angelegten Events (Tickets, Threads) wird durch den Umfang der Knoten repräsentiert; die Größe der Akteursknoten wird konstant gehalten.

[36] Bei der Mailingliste „Linux" handelt es sich um ein semi-offenes Medium, welches ausgewählten Teilen der User zur Benutzung offensteht. Es wurde in der folgenden Darstellung der Übersicht halber ausgeschlossen, ist jedoch Teil des transformierten 1-Mode-Networks und eingeschlossen in die Berechnung der Durchschnittswerte aufgrund seiner sicheren Verbindungsqualität für zwei Akteure im Netzwerk.

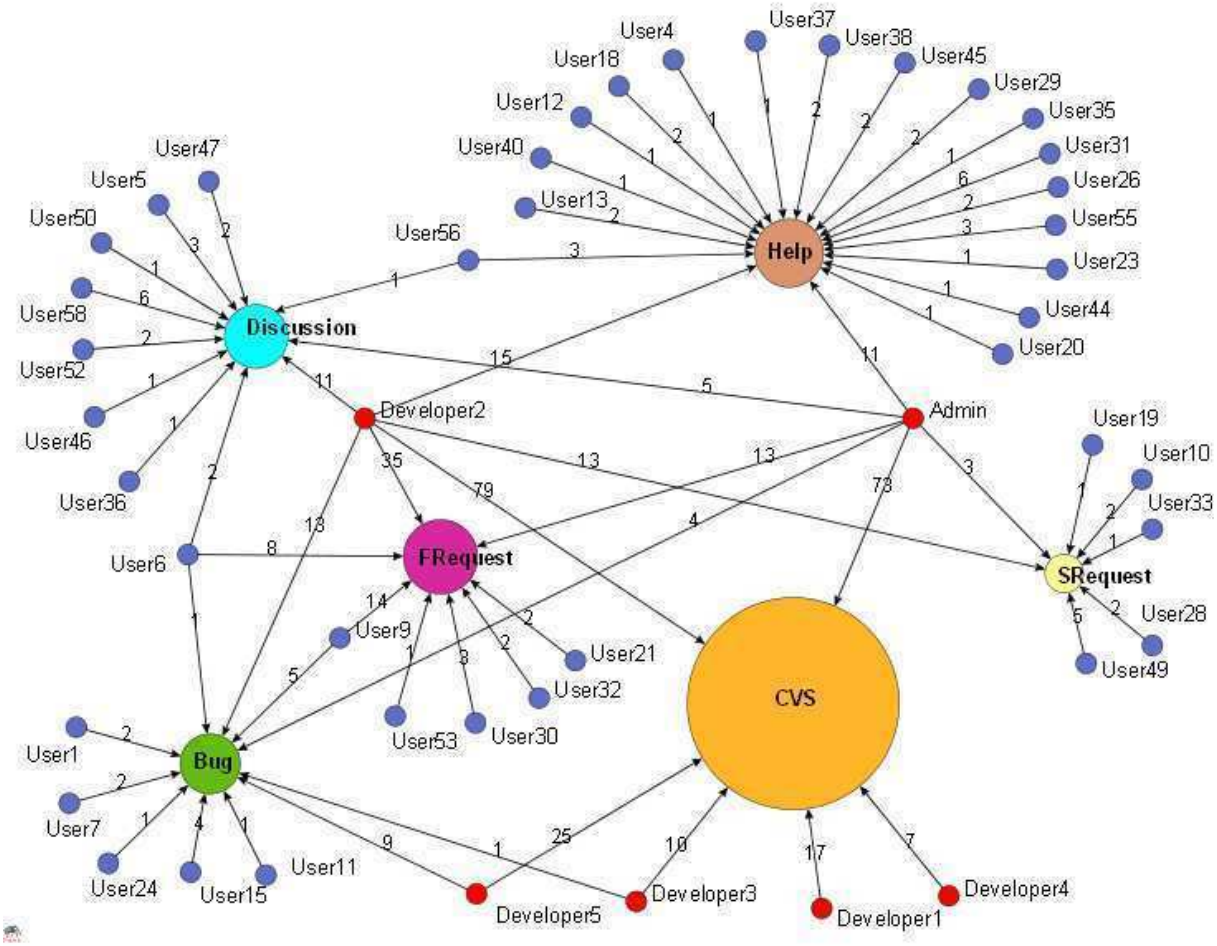


Abb. 5: Organisationstools und Beteiligungen der Akteure

Auffälliges Merkmal der vorliegenden Darstellung ist die mehrheitlich zentrierte Beteiligung der Akteure an ausschließlich einem Organisationstool. Auf 40 (38 User, 2 Developer) der 47 Akteure trifft dieser Fall zu, lediglich sieben Akteure sind mit mehr als einem Tool verbunden. Dieses Bild wiederholt sich auf der darunterliegenden Ebene der einzelnen Events (Tickets, Threads): Hier weisen lediglich drei zusätzliche Akteure (User5, Developer1, Developer4) eine Mehrfachbeteiligung auf, entsprechend den obigen Ergebnissen ausschließlich innerhalb eines Organisationstools. Damit sind auch auf dieser Ebene 37 der 47 Akteure (ausschließlich User) über nur ein Event in das Projekt eingebunden (Anhang IV). Aus diesen Ergebnissen ergibt sich ein recht deutliches Bild der ‚typischen' User-Beteiligung: Sie erfolgt überwiegend, d. h. in ungefähr 92 Prozent der Fälle, zielgerichtet zu lediglich einem Anliegen und reproduziert sich ausschließlich über diesen Fall. Eine solche Verhaltensstruktur deutet auf eine primär persönliche Problemorientierung der User hin, die ein unmittelbar sie selbst betreffendes Anliegen mitteilen und die Beziehung nach Annahme oder Ablehnung der Kommunikation beenden. Eine sich wiederholende, dem Projekt verpflichtete Beteiligung ist nicht der Regelfall.

Demgegenüber stehen die Akteure, die wiederholt bzw. mehrfach an der Kommunikation des Projekts teilnehmen. Sie setzen sich aus sechs Entwicklern und vier

Usern zusammen und zeigen eine recht große Spannweite in ihren Beteiligungsformen und -raten (Tab. 3).

Akteure	Bug	Frequest	Srequest	Discussion	Help	Linux	CVS	∑ Tools	∑ Events
User5				2				**1**	**2**
Developer4							1 (7)	**1**	**1 (7)**
Developer1							1 (17)	**1**	**1 (17)**
Developer3	1						1 (10)	**2**	**2 (11)**
User9	3	13						**2**	**16**
User56				1	1			**2**	**2**
User6	1	6		1				**3**	**8**
Developer5	3					1	1 (25)	**3**	**5 (29)**
Developer 2	5	15	5	7	10		1 (79)	**6**	**43 (121)**
Admin	4	10	3	4	6		1 (73)	**6**	**28 (101)**

Tab. 3: Beteiligungen der überdurchschnittlich aktiven Akteure im Netzwerk

Dem obigen Vorgehen folgend, lässt sich die Anzahl der Organisationstools und Events, an denen die Akteure beteiligt sind, getrennt aufgliedern und so miteinander in Beziehung setzen. Beide Dimensionen zeigen erwartungsgemäß einen deutlichen korrelativen Zusammenhang, wobei einige gesonderte Aspekte auffallen: Beteiligungen der Developer am CVS (Quellcode) erfolgen ausnahmslos mehrfach, jedoch mit sehr unterschiedlicher Beitragsanzahl und Beteiligung an den übrigen Organisationstools des Projekts. Sind Letztere gering, fallen auch die Beiträge zum Quellcode (CVS) vergleichsweise gering aus; hier besteht ein deutlicher Zusammenhang. Dabei ist jedoch zu beachten, dass die zum Quellcode beitragenden Developer dies stetig tun. Die Teilnahme an der unmittelbaren Entwicklung des Quellcodes scheint in dem Sinne ‚total' zu sein, dass ein gewisses Maß an zeitlich beständiger und aktiver Teilnahme vorausgesetzt werden kann.

Betrachtet man zudem das unterschiedliche Ausmaß der Einbindung in die verschiedenen Organisationstools, zeigt sich unter den Entwicklern eine grundlegende Differenz: Während der Admin und Developer2 in fast allen Organisationstools aktiv sind, nehmen Developer1, Developer3 und Developer4 gar nicht bzw. Developer 5 nur vereinzelt an Projektbereichen neben der Quellcodeentwicklung (CVS) teil. Auf Seiten der User zeigen sich hier deutlich weniger markante Ausprägungen, mit Ausnahme der vergleichsweise hohen und auf zwei Bereiche spezialisierten Aktivitäten

des Users9. Die unterschiedlichen Formen der Einbindung in das Netzwerk – hier insbesondere für die Entwickler – deuten im Sinne von Tätigkeitsprofilen auf arbeitsteilige Rollenstrukturen hin – ein Aspekt, den es im Rahmen der weiteren Analyse zu beachten gilt.

Im Durchschnitt entfallen ungefähr 4,7 Mitteilungen auf die Akteure und etwa 3,8 Mitteilungen auf das einzelne Event.[37] Innerhalb der einzelnen Organisationstools variieren diese Werte leicht, dies ist jedoch aufgrund der relativ kleinen Fallzahlen pro Einheit nur begrenzt belastbar. Die auffälligste Abweichung findet sich in der Einheit „Feature Request“ mit durchschnittlich 8,9 Mitteilungen pro Akteur: Unter Berücksichtigung der vorliegenden Events in diesem Bereich scheint sich hier eine verhältnismäßig kleine Anzahl an Akteuren überdurchschnittlich häufig zu engagieren (Anhang IV). Die durchschnittliche Mitteilungsanzahl pro Event zeugt von kurzen Kommunikationsprozessen, in denen die verschiedenen Anliegen „verhandelt“ werden und die, unter der Annahme der Beendigung der Kommunikation bei Erfolg, auszureichen scheinen.

5.4.3 Strukturanalyse

5.4.3.1 Zentralität

Für die Analyse des strukturellen Gefüges des Projekts wird das Netzwerk in eine 1-Mode-Variante transformiert, welche die direkten Kommunikationen ‚zwischen‘ den Akteuren abbildet. Eine solche Verbindung ist erfüllt, wenn die Akteure im gleichen Event[38] eine oder mehrere Mitteilungen getätigt haben. Trifft dies für die beteiligten Akteure bei mehreren gemeinsamen Events zu, werden die Mitteilungen der betreffenden Events summiert; im Falle eines ungerichteten, symmetrischen Netzwerkes repräsentiert der Relationswert entsprechend die Summe aller in gemeinsamen Events getätigten Mitteilungen der beteiligten Akteure. Datengrundlage der Transformation ist das von Subkomponenten und 1-In-Degree-Events bereinigte Netzwerk der vorherigen Analysen, da die Unterteilung des Netzwerkes in unverbundene Komponenten die netzwerkanalytischen Verfahren vor methodische Probleme stellt und zu Verzerrungen in den Ergebnissen führt (Jansen 2006; Trappmann et al 2011).

Das transformierte Netzwerk besteht aus 47 Knoten (den Akteuren) und 150 gerichteten Kanten, die als insgesamt 75 wechselseitige Kommunikationsbeziehungen zwischen jeweils zwei Personen vorliegen. Es kommt also im vorliegenden Netzwerk

[37] Die Beiträge zum Quellcode (CVS) wurden von der Berechnung der Durchschnittswerte ausgeschlossen, da sie ausschließlich von Entwicklern verfasst werden können und eine spezielle Sprachlichkeit (Programmiersprache) aufweisen. Mit durchschnittlich ca. 35 Mitteilungen pro Akteur zeigt sich hier im Vergleich ein deutlich höherer Wert (Anhang III).

[38] Das Organisationstool CVS wird als ein Event behandelt, die von den Entwicklern eingetragenen Revisionen werden als Kommunikationsakte bewertet.

stets zur Anschlusskommunikation. Diese Eigenschaft des Netzwerkes ist in ihrer Absolutheit ein Ergebnis der Datenbereinigung. Wie jedoch aus den obigen Untersuchungen hervorgeht, entspricht dies der tatsächlichen Kommunikationspraxis des Projekts: In nur sehr wenigen Fällen findet Kommunikation auf der Projektplattform keinen Anschluss.

Insgesamt werden nur 6,9 Prozent (Dichte) aller möglichen Verbindungen realisiert. Diese sind geprägt von dyadischen Strukturen: Etwa 51 Prozent der Akteure stehen lediglich mit einem anderen Akteur in Beziehung – ein Ergebnis, welches mit dem häufigen Auftreten einmaliger Anliegen und der geringen Anzahl beteiligter Akteure pro Event korrespondiert (Anhang IV). Die durchschnittliche Distanz zwischen den Akteuren des Netzwerkes weist mit 2,1 Kommunikationsschritten und einer maximalen Distanz von 3 Kommunikationsschritten eine recht geringe Ausprägung auf, wobei die Akteure in die Kommunikationen des Netzwerkes relativ ungleich eingebunden sind. Solche Strukturmerkmale lassen sich mit Konzepten der Zentralität ausdrücken, welche anhand spezifischer Relationsqualitäten die Einbindung der einzelnen Knoten erfasst und in Bezug zur größtmöglichen Ausprägung dieser Qualität im gegebenen Netzwerk setzt. Die Unterschiede der einzelnen Punktzentralitäten der Knoten lassen sich dann in Summe auf die maximal erreichbare Zentralisierung (max. Abweichung der Punktzentralitäten) des vorliegenden Netzwerkes beziehen und als globales Strukturmerkmal für das gesamte Netzwerk zwischen 0 (keine Zentralisierung) und 1 (maximale Zentralisierung) standardisiert datieren (Jansen 2006; de Nooy et al. 2011).

Die drei wichtigsten Zentralitätskonzepte – (1) Degree-, (2) Closeness- und (3) Betweenness-Zentralität – basieren auf unterschiedlichen Relationskriterien und entsprechenden qualitativen Zielsetzungen: Die Degree-Zentralität (1) misst die Anzahl der direkten Verbindungen zu anderen Knoten des Netzwerkes und repräsentiert die Kommunikationsaktivität der Knoten, während die Closeness-Zentralität (2) als Maß der (Un-)Abhängigkeit sowie der Effizienz auf die Nähe eines Knotens zu allen anderen Knoten zurückgreift. Als Maß für die mögliche Kommunikationskontrolle erfasst die Betweenness-Zentralität (3) die Anzahl der kürzesten Verbindungen zwischen Punktpaaren, die über den jeweils betrachteten Knoten laufen (Jansen 2006). Alle drei Kennwerte[39] weisen als globale Strukturmerkmale mit jeweils über 0.7[40] eine recht hohe Ausprägung auf und belegen ein hohes Maß struktureller Ungleichheiten der Knoten. Die Verteilungen der jeweiligen Zentralitätswerte der einzelnen Knoten

[39] Der Closeness-Zentralität kommt aufgrund der sehr geringen allgemeinen Spannweite der Distanzen im Netzwerk eine untergeordnete Bedeutung zu.
[40] Degree-Zentralisierung: 0.77, Closeness-Zentralisierung: 0.74, Betweenness-Zentralisierung: 0.73.

sowie die Überprüfung des Netzwerkes auf Cutpoints[41] erlauben in diesem Rahmen die Identifizierung substantieller Knoten: Dabei weisen insbesondere der Admin und Developer2 eine herausgehobene Bedeutung für die Kommunikation auf, sowohl in der direkten Beteiligung (Degree) als auch in ihrer Qualität als Vermittler (Betweenness) (Anhang IV). Für einen nicht geringen Anteil der Knoten stellen sie zudem einen Cutpoint dar und fungieren im Falle des Admins für sieben Akteure und des Developer2 für 17 Akteure als alleinige Anschlussstelle (Anhang V).

Vor dem Hintergrund des hier zugrunde gelegten theoretischen Rahmens, der analytisch primär auf das Relationsgefüge abzielt, lässt sich demgemäß zunächst nur auf besonders relevante Kommunikationsstellen in der Projektorganisation schließen. Eine umfassende Einschätzung der Bedeutung und Funktion der Knoten bzw. Stellen ist jedoch erst in Beziehung zum Strukturgefüge im Sinne einer relationalen Gesamtpositionierung möglich. Im Folgenden wird daher im Anschluss an die netzwerktheoretische Konzeption sozialer Systeme von Fuchs (2005) das Kommunikationsnetzwerk zunächst auf Cliquen und strukturelle Äquivalente hin analysiert, um auf dieser Grundlage dann die Einschätzung spezifischer Systemdifferenzierungen und entsprechender Positionsgefüge aufzubauen.

5.4.3.2 Cliquen

Das Konzept der Clique basiert auf maximal kompletten Beziehungsstrukturen unter *n* Akteuren, d. h. der Verbindung aller Beteiligten untereinander. Die kleinste Clique ist die vollständige Triade, die unter dem Kriterium maximaler Verbundenheit in Bezug auf die Anzahl der beteiligten Akteure erweiterbar ist. Dabei gilt: Cliquen mit größerer Anzahl an Akteuren beinhalten stets die jeweilige Anzahl an Cliquen mit kleinerer Anzahl an Akteuren (de Nooy et al. 2011). Die Analyse der Netzwerkstrukturen auf kohäsive Zusammenhänge zielt daher zunächst auf das größtmögliche Gefüge kompletter Beziehungsstrukturen, um dann kleinere Ausprägungen in ihrer relativen Verbundenheit differenziert betrachten zu können. Die Verfahren operieren auf der Grundlage eines ungerichteten, binären Netzwerkes, da sie auf die realisierten Relationsausprägungen ohne deren numerische Qualität zielen.

Für das vorliegende Netzwerk ergibt sich dabei folgendes Bild: Die größte, vollständige Clique bildet eine Hexade und setzt sich aus allen aktiven ‚formalen' Mitgliedern des Projekts – dem Admin und allen Developern – zusammen. Diese Formation erklärt sich überwiegend aus der gemeinsamen Beteiligung an der Entwicklung des Quellcodes und den entsprechenden Kommunikationsbeziehungen über das CVS-Tool. Die Gruppe weist die vollständigste interne Verbundenheit auf und bildet insofern eine reziproke Formation maximaler Qualität. Zudem lassen sich insgesamt vier

[41] Cutpoints sind Knoten, deren Wegfall das Netzwerk in einzelne unverbundene Komponenten aufteilen oder einzelne isolierte Knoten abtrennen würde.

Tetraden und zehn Triaden maximaler Verbundenheit identifizieren, deren sich zum Teil überschneidende Mitglieder weitere Cliquengefüge ausbilden (Tab. 4).

Cliquen	Akteure	Größe
1.	Developer1, Admin, Developer2, Developer3, Developer4, Developer5	6
2.	Admin, User9, Developer2, Developer5	4
3.	Admin, User6, User9, Developer2	4
4.	Admin, Developer2, User18, User29	4
5.	Admin, User9, Developer2, User30	4
6.	Admin, User5, Developer2	3
7.	Admin, Developer2, User15	3
8.	Admin, Developer2, User19	3
9.	Admin, Developer2, User31	3
10.	Admin, Developer2, User32	3
11.	Admin, Developer2, User47	3
12.	Admin, Developer2, User56	3
13.	User4, Developer2, User44	3
14.	User9, User11, Developer2	3
15.	User13, Developer2, User26	3

Tab. 4: Cliquenformationen im Netzwerk

Auf der Basis der gemeinsamen Beteiligung der Akteure an den (verschiedenen) Cliquen lassen sich über hierarchische Clusterverfahren diese Strukturausprägungen nach ihrer Ähnlichkeit schrittweise zusammenfassen und zu einem differenzierten Bild kohärenter Relationsgefüge ausarbeiten: Auf den höchsten Ähnlichkeitsniveaus lassen sich die Akteure Admin, Developer2 und User9 sowie folgend Developer1, Developer3, Developer4 und Developer5 in einem ersten Schritt als Cluster zusammenfassen (Anhang VI). Legt man nun als Basiseinheit des Verfahrens die vollständige Triade zugrunde, werden der Gruppierung als nächstähnliche Akteure User6 und User30 zugeordnet, und es kommt zu drei separaten Cliquenformationen der Knoten User18 und User29 (Fall 1), User44 und User4 (Fall 2) sowie User13 und User26 (Fall 3) (Anhang VI). Während die ‚Separierung' in Fall 2 und Fall 3 in den verschiedensten Verfahren bis zu einem sehr hohen Toleranzmaß bestehen bleibt, zeigen sich in Fall 1 verschiedene Zuordnungsmuster in Abhängigkeit vom gewählten

Verfahren. Wählt man eine kleine Grundeinheit der Cliquenberechnung, erfolgt ihre ‚Eingliederung' erst auf einem geringen Ähnlichkeitslevel. Zielt man jedoch methodisch auf größere strukturelle Relationsgefüge oder zieht man die Bedeutung der Knoten als essentieller Träger für die Gesamtstrukturen in Betracht, verringern sich die Unterschiede (Anhang VI). Die Gruppe der Knoten, die ausschließlich über Triaden in das Netzwerk eingebunden sind, weist schließlich die geringsten[42] strukturellen Überschneidungen mit den übrigen Clustern auf und kann als eine Vielzahl von Einzelformationen betrachtet werden.

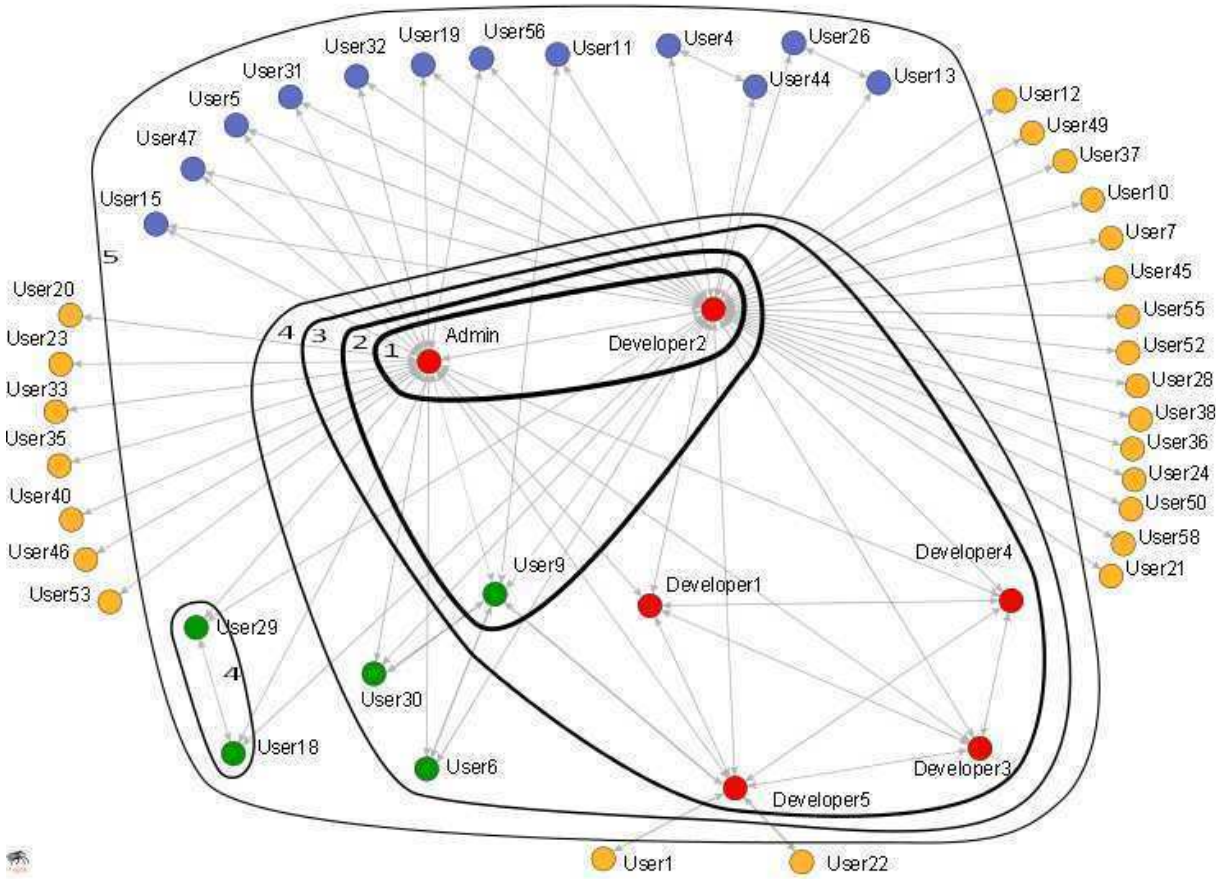


Abb. 6: Knoteneinbindung in Cliquen und approximierte Clusterschritte als Isograph

Im vorliegenden Netzwerk lassen sich eine Reihe von Cliquen unterschiedlicher Qualität (Triaden, Tetraden, Hexade) identifizieren, die als solche zunächst eigenständige Gefüge maximaler ‚interner' Verbundenheit repräsentieren (Tab. 4). Diese Formationen weisen jedoch recht ausgeprägte Überschneidungen in den sie tragenden Knoten auf, so dass ihre externe Abgrenzung nur bedingt gegeben ist. Die Verfahren der hierarchischen Clusteranalyse zeigen kaum separierte kohäsive Gefüge, die sich auf einem akzeptablen Kongruenzniveau als solche differenzieren lassen. Die deutlichste Ausprägung zeigt in diesem Zusammenhang die Gruppe der Entwickler, die als Hexade über eine vollständige reziproke Verbundenheit verfügt, jedoch über Admin

[42] Das Niveau der Zuordnung variiert für einige wenige Knoten je nach Einschätzung des vorherigen Aspektes (Fall 1) und der entsprechenden Auswahl eines Verfahrens.

und Developer2 sowie bedingt Developer5 auch intensive Außenverbindungen aufweist. In Abbildung 6 repräsentiert die Farbgebung der Knoten die maximale Größe der Cliquenformation(en), in welche die einzelnen Knoten eingebunden sind (vgl. auch Tab. 4). Ihre Aggregation in ähnliche Gruppen zeigen die verschiedenen Mengenkreise, die im Sinne von Isolinien näherungsweise die Clusterschritte abnehmender Ähnlichkeit der angewandten Verfahren zusammenfassen. Anhand dieser Darstellung lässt sich gut erkennen, dass die Gruppierungen verschachtelt sind und im Grunde ausgehend von der Entwicklergruppe ein einziges, immer größer werdendes Cluster repräsentieren. Diese Ergebnisse deuten auf eine grundlegende Zentralisierung des Netzwerkes hin, die nur bedingt abgeschlossene Cliquenformationen hervorbringt. Damit rückt die Ähnlichkeit der Knoten aufgrund ihrer strukturellen Position im gesamten Netzwerkgefüge, welche sich über den globalen Vergleich der Relationsmuster erschließen lässt, in den Fokus.

5.4.3.3 Strukturelle Äquivalenz

Das Konzept der strukturellen Äquivalenz gründet auf dem Vergleich der Beziehungsmuster von Knoten, die im Gegensatz zur Cliquenanalyse nicht notwendig miteinander verbunden sein müssen, und sucht anhand der Ähnlichkeit der vorliegenden Beziehungsmuster nach äquivalenten Positionen in den Strukturen des Netzwerkes. Diese lassen sich dann aus einer globalen Perspektive mit spezifischen Funktionseigenschaften für das Netzwerk verbinden und als Rollenmuster interpretieren (Jansen 2006). Dabei lassen sich drei unterschiedliche Konzepte von Äquivalenz unterscheiden[43]:

Strukturell äquivalent (1) sind Knoten, die zu identischen (dritten) Akteuren dieselben Verbindungen aufweisen und deren Austausch insofern keine Auswirkungen auf die Struktur des Netzwerkes als auch das Beziehungsprofil der betroffenen Knoten haben würde. Dies trifft im Netzwerk der Abb. 7 für die Knoten E und F sowie H und I zu, während die restlichen Knoten jeweils eigenständige Klassen darstellen. Schwächt man das Kriterium der Beziehung zu identischen Knoten ab und greift stattdessen auf identische Beziehungs*profile* der Knoten zurück, lassen sich diese zu automorph-äquivalenten Klassen (2) zusammenfassen. Diese Gruppierungen bestehen aus Knoten, die bei Austausch untereinander und mit den Knoten der anderen Klassen zu keinen Veränderungen der Distanzen aller Knoten im Netzwerk zueinander führen. Unter diesen Bedingungen ergeben sich im Beispielnetzwerk fünf Klassen automorpher Äquivalenz: {A}, {B,D}, {C}, {E,F,H,I} und {G}. Das Konzept der regulären Äquivalenz (3) abstrahiert dagegen sowohl von der Identität der Beziehungs-

[43] Die Unterteilung der Konzepte struktureller Äquivalenz und das vorliegende Beispielnetzwerk sind den Ausführungen zur UCINET-Software von Hanneman und Riddle (2005) entnommen. Die Begriffe „Klasse“ und „Block“ werden dabei im Folgenden synonym verwendet und bezeichnen aggregierte Mengen ähnlicher Knoten bzw. Positionen ohne jede ideologische Bedeutung.

knoten als auch von der Anzahl der Beziehungspartner – d. h. auch von identischen Beziehungsprofilen – und betrachtet Knoten als äquivalent, die über gleiche Beziehungen zu anderen Knoten verfügen, die selbst wiederum eine regulär äquivalente Klasse sind. Angewendet auf das Beispielnetzwerk lassen sich unter diesen Voraussetzungen die Knoten {A}, {B,C,D} und {E,F,G,H,I} als regulär äquivalente Klassen zusammenfassen. Wie schon in der zirkulären Definition erkennbar, erfolgt die Bestimmung dieser Form äquivalenter Knoten stets in Abhängigkeit von in Beziehung stehenden Knoten, die selbst als regulär äquivalent zusammengefasst werden müssen. Da deren Zuordnung zu äquivalenten Klassen jedoch wiederum von den Knoten abhängt, mit denen sie – eingeschlossen erstere – in Verbindung stehen, lassen sich entsprechend vielfältige und keine endgültigen Lösungen finden.

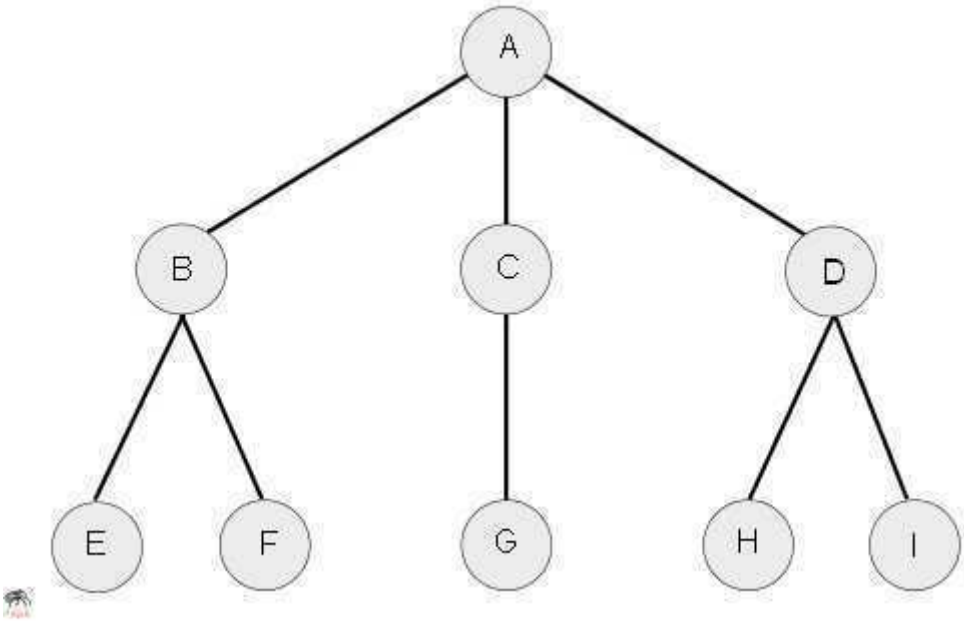


Abb. 7: Beispielnetzwerk für unterschiedliche Äquivalenzqualitäten

Strukturell und regulär äquivalente Klassen von Knoten können aufgrund ihrer vollständig vorhandenen oder fehlenden Verbindungen zu anderen Knoten, die dann selbst wiederum Klassen darstellen, als Blöcke zusammengefasst werden. In einer entsprechenden Image-Matrix repräsentieren dann die einzelnen Zeilen und Spalten nicht mehr die Knoten, sondern die aggregierten Blöcke bzw. Klassen, und deren Überschneidungen geben die Beziehungsform zwischen den jeweiligen Blöcken an, wobei idealtypisch zwischen vorhandenen strukturellen oder regulär äquivalenten Beziehungen und keinerlei Beziehungen unterschieden wird[44]; die Diagonale repräsentiert die Beziehung der zusammengefassten Knoten untereinander. Im Vorder-

[44] Mit zunehmender Abschwächung der Äquivalenzkriterien entfällt natürlich die Garantie, dass alle Knoten einer Position *k* die gleichen Verbindungen zu den Akteuren der Position *l* haben. Hier müssen zusätzliche Kriterien (z. B. die Dichte) herangezogen werden, um über die Annahme einer Verbindung zu entscheiden (vgl. Trappmann et al. 2011; Jansen 2006).

grund des Verfahrens steht entsprechend das Relationsmuster der verschiedenen Positionsblöcke zueinander, um grundlegende Strukturmerkmale und -prinzipien des Netzwerkes zu gewinnen (Jansen 2006). Verbindungen zwischen den Knoten, welche der Blockeinteilung und dem zugrunde gelegten (idealtypischen) Relationsmuster widersprechen, werden in ihrer Anzahl als Fehlerquote ausgegeben und dienen als Qualitätsmaß der Einteilung.

Methodisch stellt sich dabei das Problem, dass sich die jeweiligen Positionen relational gegenseitig bestimmen und die Identifizierung äquivalenter Positionen und Blöcke aus einer Vielzahl potenzieller Gruppierungs- und Zuordnungsmöglichkeiten folgt, welche mit wachsender Größe des Netzwerkes erheblich zunehmen (Trappmann et al. 2011). Das Vorgehen sollte daher zunächst auf die Gruppierung möglichst ähnlicher Knoten abzielen, um dann in entsprechenden Blockmodeling-Verfahren optimiert werden zu können.

Um eine umfassende Grundlage für den Vergleich der Beziehungsprofile der Knoten im Netzwerk zu erhalten, wurde auf eine Vielzahl möglicher Verfahren zurückgegriffen, die sowohl auf Ähnlichkeits- als auch auf Ungleichheitsmaße zielen und diese als Berechnungsgrundlage für den Prozess des Clusterings heranziehen. Dabei ergaben sich für beide Maße recht ähnliche Ergebnisse, die sich aus ihrer jeweiligen Zuordnungslogik der Knoten zu abgrenzbaren Gruppen ergänzten (ausführlich Anhang VII). Es lassen sich zusammenfassend als „Durchschnittsvariante“ insgesamt acht Gruppierungen unterteilen, die sich wie folgt zusammensetzen:

Cluster	Knoten
A	Admin
B	Developer2
C	Developer1, Developer3, Developer4, Developer5
D	User6, User9, User11, User30
E	User5, User15, User18, User19, User29, User31, User32, User47, User56
F	User1, User22
G	User20, User23, User33, User35, User40, User46, User53
H	User4, User7, User10, User12, User13, User21, User24, User26, User28, User36, User37, User38, User44, User45, User49, User50, User52, User55, User58

Tab. 6: Clustereinteilung auf der Grundlage von Ungleichheitsmaßen der Verbindungsprofile (Anhang VII: Pajek Dissimilarity)

Eine solche Einteilung hat nur vorläufigen Charakter; sie enthält zwar Informationen über die grundlegende Ähnlichkeit der Verbindungsprofile der Knoten, nicht jedoch zur spezifischen Form der Ähnlichkeit und Verschiedenheit der Knoten in Beziehung

zu den anderen Knoten und Knotenklassen des Netzwerkes (Hanneman/Riddle 2005). Hierzu dient das Verfahren des Blockmodelings, welches mit dem Ziel der Identifizierung eines zusammenfassenden Strukturierungsmusters seine Anwendung findet. Diese Vorgabe macht ein abwägendes Vorgehen notwendig, um einen Ausgleich zwischen idealtypischem Strukturmodell und einer Klassenaufteilung mit möglichst geringer Fehlerquote zu finden. Auf der Grundlage der bisherigen Ergebnisse orientiert sich die Modellierung dabei an der Arbeitshypothese eines Zentrum-Peripherie-Verhältnisses, die es durch schrittweise Variation der obigen Klasseneinteilung und anschließende Berechnung zu überprüfen gilt. Bisher gewonnenes Wissen über die Rahmenbedingungen der Projektorganisation sowie über die strukturellen Eigenschaften des Netzwerkes sollten und werden in diesen Prozess eingebunden. Das ausführliche Vorgehen und die entsprechenden Ergebnisse der Blockanalyse finden sich in Anhang VIII; die folgenden Ausführungen wählen eine die wesentlichen Schritte zusammenfassende Darstellung.

Prüft man die obige Klasseneinteilung als Blöcke auf ihr Verhältnis sowohl intern als auch in Beziehung zu den anderen Blöcken, erhält man ein relativ deutliches Bild mit einer recht geringen Fehlerquote von 14 abweichenden Verbindungen aus insgesamt 150 gerichteten Beziehungen: Die Klassen „Admin“ und „Developer2“ verfügen über vollständige Beziehungen untereinander und zum Großteil der anderen Blöcke, welche ihrerseits keinerlei Verbindungen zueinander aufweisen. Zudem zeigen intern lediglich Klasse C komplette und Klasse D reguläre Relationsstrukturen, die übrigen Blöcke verfügen über keine internen Verbindungen bzw. erhalten aufgrund ihrer Geringfügigkeit in Klasse E und F den Status der Fehlverbindung (Anhang VIII). Aufgrund der ähnlichen Beziehungsmuster der Knotenblöcke „Admin“ und „Developer2“ wurden diese in einem ersten Schritt als ein Block zusammengelegt, ebenso wie die Klassen F, G und H, die sowohl in ihrer Degree-Verteilung als auch in der visuellen Darstellung des Netzwerkes erkennbar große Gemeinsamkeiten aufweisen (Kap. 5.4.2; Abb. 6).

Die Fusion der Blöcke verringert deren Anzahl von acht auf fünf Klassen, die wiederum der Blockmodellanalyse unterzogen wurden. Mit einer Fehlerquote von 20 abweichenden Verbindungen zeigt diese Einteilung zwar eine etwas schlechtere ‚Passung‘, die jedoch vor dem Hintergrund der deutlichen Reduktion der Klassenanzahl akzeptabel ist. Der fusionierte Block 1, bestehend aus Admin und Developer2, behält auch in dieser Einteilung seine kompletten bzw. regulären Verbindungsstrukturen zu Klasse C und E bzw. Klasse D und steht in einem Verhältnis regulärer Äquivalenz zum aggregierten Block 5 (Anhang VIII). Die Image-Darstellung zeigt dabei eine deutliche Ähnlichkeit von Block 5 und Klasse E, welche in gleichen Relationen zu den übrigen Blöcken stehen und sich lediglich in der Qualität der Beziehungen unterscheiden. Fusioniert man auch diese Blöcke – d. h. Klasse A und B sowie E, F, G

und H werden zusammengefasst – bleibt die Fehlerquote konstant und die gemeinsame, einzig bestehende Verbindung zu Block 1 prägt eine reguläre Äquivalenzqualität aus (Anhang VIII). Betrachtet man auf dieser Stufe die vorliegenden Fehlerausprägungen genauer, fallen vor allem jene Posten ins Gewicht, die gegen das Nullblockkriterium[45] der umgruppierten Soziomatrix verstoßen (Jansen 2006). Im vorliegenden Fall sind diese ausschließlich durch Developer5 verursacht und lassen sich durch dessen Zuordnung zu Block 1 „beseitigen". Die sich daraus ergebende Klasseneinteilung verfügt mit sechs abweichenden Verbindungen über eine sehr geringe Fehlerquote, wobei die Fehler sich zudem ausschließlich über vereinzelte Verbindungen von Knoten innerhalb des vierten Blocks erklären lassen (Abb. 8).

Mittels der Verfahren der Blockanalyse lässt sich die Einteilung der Knoten nicht weiter optimieren, lediglich Klasse C und D könnten ggf. bei konstanter Fehlerquote fusioniert werden. Allerdings würde dieser Schritt zu Minderungen im Äquivalenzniveau der Relationen zwischen als auch innerhalb der Blöcke führen und entspräche auch nicht den Unterschieden der Blockmitglieder bezüglich ihres (formalen) Status. Die grundlegende Ähnlichkeit der Knoten dieser beiden Blöcke ist also zwar festzuhalten, jedoch scheint aus Gründen der Plausibilität die Unterteilung des Netzwerkes in vier Blöcke die zu bevorzugende Lösung zu sein.

	1	**2**	**3**	**4**
1	com	com	reg	reg
2	com	com	–	–
3	reg	–	reg	–
4	reg	–	–	–

	1	**2**	**3**	**4**
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	6

Abb. 8: Ergebnis der endgültigen Blockeinteilung als Image-Matrix mit Fehlermatrix

Die Image-Matrix der gewählten Partitionierung zeigt vier Klassen, die sowohl in Relationen unterschiedlicher Qualität zueinander stehen als auch intern verbunden sind. Der erste Block umfasst drei strukturell äquivalente Positionen, die sich über *identische* Beziehungen untereinander und zu den Mitgliedern des zweiten Blocks – den restlichen Entwicklern – auszeichnen. Diese Übereinstimmungsmerkmale der Beziehungsprofile liegen auch für Block 2 vor, sowohl zwischen den Knoten innerhalb des Blocks als auch in Relation zu Block 1. Zur Gruppe von User6, User9, User11 und User30 als drittem Block und zu den restlichen Usern im vierten Block verfügen lediglich die Mitglieder der ersten Klasse über Verbindungen, die aggregiert jeweils einer regulär äquivalenten Qualität entsprechen. Das heißt, die Beziehungen der entspre-

[45] Das Nullblockkriterium steht für die Trennung von Blöcken unter der Bedingung vollständig fehlender Beziehungen zwischen den Knoten der Blöcke. Seine Bedeutung fußt auf theoretischer Ebene auf dem Postulat der Konstitution von sozialen Strukturen gerade aufgrund von Beziehungsbarrieren, während es methodisch im Rahmen der gegenseitigen Bestimmung der Blöcke und Positionen eine besondere Relevanz hat (Jansen 2006).

chenden Klassenknoten zueinander sind vollständig im Sinne einer mindestens einmaligen Bezugnahme zwischen zwei Knoten unterschiedener Blöcke, und zwar für jeden Knoten eines Blocks mindestens einmal, darüber hinausgehend jedoch ohne Beachtung der Verteilung bzw. Ähnlichkeit der Beziehungsprofile der Knoten. Dies gilt auch für die internen Beziehungen der Mitglieder der dritten Klasse, ansonsten weisen die Blöcke 2, 3 und 4 keine weiteren Beziehungsstrukturen auf.

Die Beziehungen der Klassen zueinander und auch deren unterschiedliche Qualität hinsichtlich der Äquivalenz bestätigen deutlich die grundlegende Strukturierung des Netzwerkes nach einem Zentrum-Peripherie-Prinzip. Der Kern, bestehend aus Admin, Developer2 und Developer5, unterhält als einzige Klasse Beziehungen zu allen anderen Blöcken, die ihrerseits über keine Verbindungen untereinander verfügen. In diesem Zusammenhang sind die Unterschiede im Niveau der Äquivalenz der Relation sowohl zwischen als auch innerhalb der Blöcke von besonderer Bedeutung: Die 37 Knoten des vierten Blocks weisen intern (annähernd) keine Beziehungen auf und schließen unter den Bedingungen regulärer Äquivalenz in ihren 46 Verbindungen ausschließlich an die drei Mitglieder des Zentrums an. Die gleiche Form der Einbindung in die Netzwerkstrukturen liegt bei Block 3 vor, jedoch ist hier die durchschnittliche Anzahl der Verbindungen der Akteur zum Zentrum höher (vier Akteure, acht Verbindungen), und diese Formation grenzt sich als eigenständige Klasse über ihre internen Beziehungsstrukturen ab. Die beiden Blöcke repräsentieren damit fast idealtypisch das Verhältnis einer Peripherie (Block 4) und Semi-Peripherie (Block 3) zu einem Zentrum (Block 1).

Etwas abweichend hiervon stellt sich Block 2 dar: Zwar weisen die Mitglieder dieser Klasse ebenso wie Block 3 und 4 externe Beziehungsstrukturen ausschließlich zu Block 1 auf, jedoch zeigen sie in ihrer Anbindung an das Zentrum und ihren internen Beziehungsstrukturen das Maximum gegenseitiger Verbundenheit. Damit stellen sie eine sehr geschlossene Formation dar, bilden allerdings in Rückgriff auf die Cliquenanalyse mit den Mitgliedern des ersten Blocks eine vollständige Hexade und sind insofern nur schwer von dieser zu trennen. Ihre Einordnung in das Kontinuum zwischen Zentrum und Peripherie bereitet daher zunächst einmal Schwierigkeiten. Vor dem Hintergrund des Umstandes, dass es sich bei dieser Gruppe jedoch vollständig um eingetragene Entwickler des Projekts handelt und die Mitglieder von Block 2 sich ausschließlich an der direkten Entwicklung des Quellcodes beteiligen, ist dieser Block viel eher als eine hinter dem Zentrum liegende ‚Subeinheit' zu verstehen. Peripherie und Semi-Peripherie bestehen schließlich aus Usern, die über die verschiedenen Tools ihre Anliegen bezüglich der produzierten Software mitteilen, die von den Developern des Projekts entwickelt wird. Letztere bilden insofern den gemeinsamen Kern des Projekts, jedoch unterscheiden sich die Developer des ersten und des zweiten Blocks durch die völlige Abschottung der letzteren von den Usern.

Diese Strukturausprägungen eines Zentrum-Peripherie-Verhältnisses lassen sich mittels einer permutierten Netzwerkmatrix verdeutlichen, in welcher die Zeilen und Spalten die Akteure in identischer Reihenfolge repräsentieren und die schwarze Markierung der einzelnen Matrixelemente eine Verbindung der entsprechenden Akteure anzeigt (Abb. 9). Admin, Developer2 und Developer5 fungieren auffällig als das kommunikative Zentrum des Netzwerkes, welches als Block zu allen anderen Akteuren des Netzwerkes in Verbindung steht. Sie bilden mit den restlichen Entwicklern (links oben) eine Formation maximaler Verbundenheit und lassen sich insofern als Clique zusammenfassen. In abgeschwächter Form besteht ein solches (äquivalentes) Verhältnis auch zur Semi-Peripherie (User6, User9, User11, User30), die jedoch ‚intern' vor allem über User9 in Kontakt zueinander stehen. Die Peripherie zeigt mit Ausnahme der sechs Abweichungen im Gegensatz dazu keinerlei interne Kontakte und ist weit von einer Schließung entfernt.

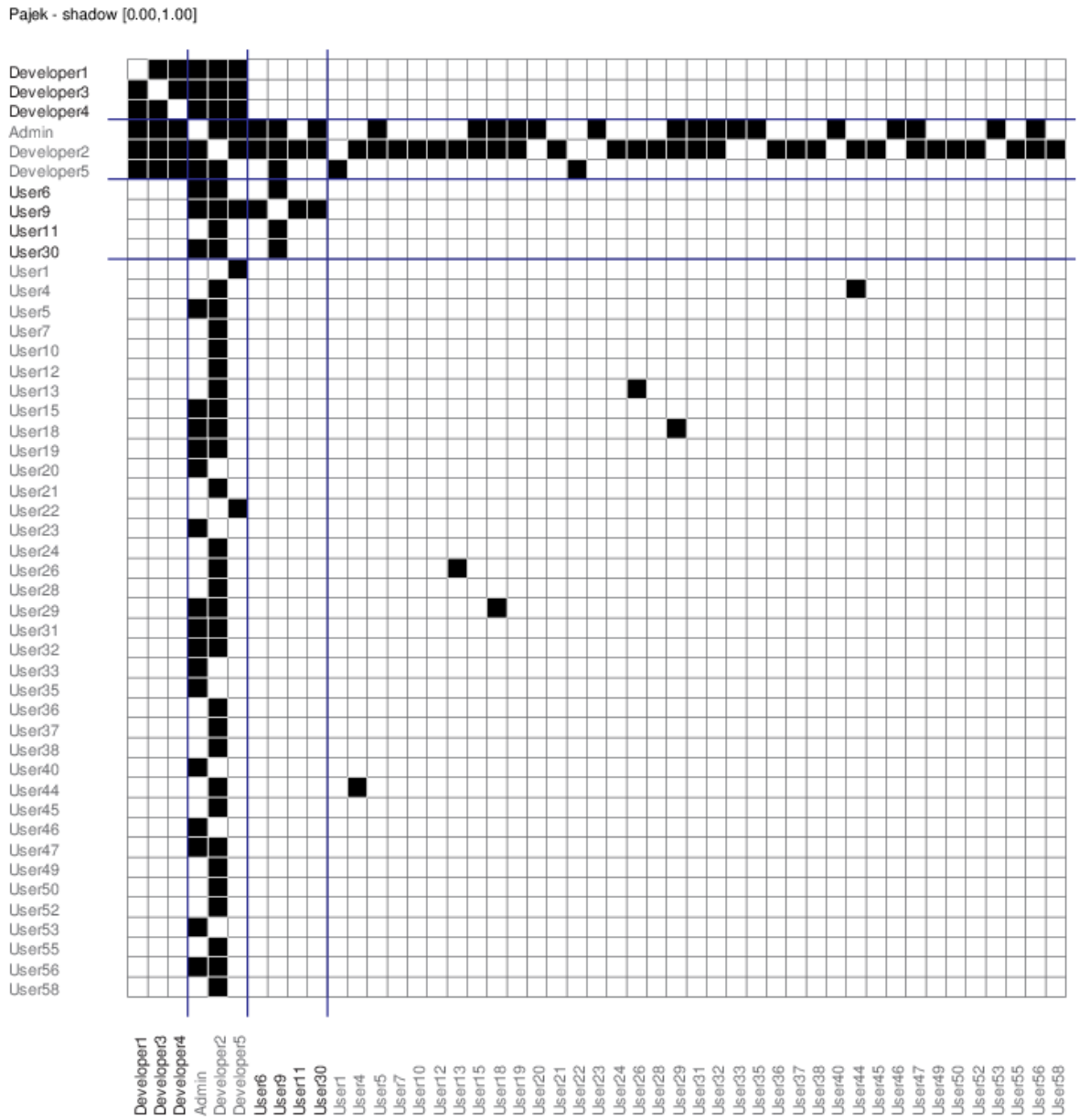


Abb. 9: Netzwerkmatrix, permutiert nach Blockeinteilung

5.4.3.4 Position und Rolle

Die Einsichten in das grundlegende Strukturierungsprinzip des Netzwerkes lassen sich vor dem Hintergrund der Klasseneinteilung noch erweitern, wenn man die Einbindung der einzelnen Knoten in ihre lokalen Strukturen berücksichtigt. Ein solcher Ansatz zielt auf die Funktion der Knoten*positionen*, welche sich aus strukturellen Beschränkungen der Kommunikationswege in Abhängigkeit von der Zugehörigkeit der Knoten zu Cliquen oder Klassen ergeben. An die jeweiligen Strukturausprägungen lassen sich dann *Rollen*konzepte anschließen, die den Knoten aufgrund ihrer relationalen Position zugeschrieben werden (Jansen 2006).

Für das vorliegende Netzwerk sind aufgrund der vollständigen Symmetrie der Verbindungen vier unterschiedliche Rollenmuster möglich: Coordinator (1), Gatekeeper/Representant (2), Itinerant Broker (3) und Liaison (4). Diese Rollenkonzepte basieren auf der Ausprägung einer unvollständigen Triade und lassen sich sehr einfach anhand eines Beispielnetzwerkes veranschaulichen:

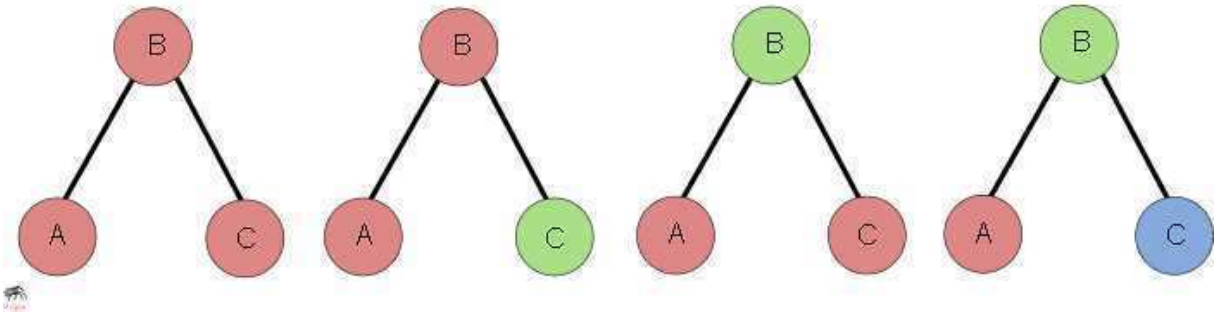


Abb. 10: Unterschiedliche Brokerkonstellationen (Akteur B) in einer unvollständigen Triade

Rollenträger in allen Konstellationen ist stets der Knoten – hier: Knoten B –, der zwischen zwei anderen, nicht miteinander verbundenen Knoten im Sinne eines Vermittlers steht; die Farbgebung in Abb. 10 spiegelt die ‚Gruppenzugehörigkeit' der Knoten wider. Innerhalb einer Gruppe oder Klasse legt man bei einem solchen Strukturgefüge den Schwerpunkt der Funktion auf die Koordinationsmöglichkeiten dieser Position (1). Sind hingegen zwei verschiedene Gruppen oder Klassen beteiligt, ergeben sich je nach Zugehörigkeit des Vermittlerknotens zwei unterschiedliche Muster: Weisen Knoten A und B die gleiche Zugehörigkeit auf, fungiert Knoten B (je nach ‚Kommunikationsrichtung') als Repräsentant oder Gatekeeper seiner Gruppe bzw. Klasse (2). Sind jedoch seine beiden Verbindungsknoten zusammengehörig, bezeichnet man seine Rolle als Itinerant Broker oder auch Consultant (3). Die Besonderheit dieser Position liegt in der Vermittlung von Informationen an eine ‚fremde' Gruppe, die er selbst aus dieser Gruppe an anderer Stelle erhalten hat. Sind schließlich alle Knoten Mitglieder unterschiedlicher Gruppen oder Klassen, fungiert Knoten B als ‚reiner' Vermittler; der Ausdruck „Liaison" betont hierbei die Bindung zweier Knoten durch einen Dritten (4) (de Nooy et al. 2011).

Um ein möglichst vollständiges Bild der Brokerpositionen für das Kommunikationsnetzwerk des Projekts zu erhalten, wurde die Analyse auf der Grundlage verschiedener Knoteneinteilungen vorgenommen: der Unterscheidung zwischen Developer und User, der Klassenzuordnung der Blockanalyse und einer modifizierten Version der Klasseneinteilung, die den obigen Ausführungen folgend alle Developer und den Admin des Projekts zu einem Block zusammenfasst. Die jeweiligen Ergebnisse zeigen dabei nur wenige Unterschiede und erlauben die Identifikation grundlegender Positionen im Netzwerk (Anhang IX). Auf Seiten der User zeigt lediglich User9 be-

sondere Positionsqualitäten: Er fungiert ebenso als Gatekeeper/Representant wie als Coordinator. Die geringe Ausprägung dieser Positionsqualitäten sowie die Konstanz der Ergebnisse in Bezug auf die verschiedenen Knoteneinteilungen zeigen jedoch, dass diese Qualitäten nur in Bezug auf die Mitglieder der Semi-Peripherie Geltung besitzen. Unter den Entwicklern des Projekts nehmen der Admin, Developer2 und Developer5 – also Block 1 – ausgeprägte Gatekeeper/Representant-Positionen ein und fungieren zudem als Itinerant Broker für die User. In Bezug auf die Klasseneinteilung der Blockanalyse verlieren diese Positionseigenschaften an Qualität, dafür lässt sich ihnen jetzt auch Liaison-Charakter attestieren – ein Umstand, der letztlich auf die Unterteilung der Knoten in eine höhere Anzahl an Klassen zurückzuführen ist (Anhang IX).

Eine abschließende Einschätzung dieser Ergebnisse ist erst vor dem Hintergrund des organisationalen Kontextes des Projekts und mit Bezug auf die globale Struktur des Netzwerkes zu leisten. Die Ergebnisse für User9 belegen die Abgeschlossenheit von Block 3, der intern vernetzt ist und nach ‚außen' hin in ausschließlichem Kontakt mit dem Zentrum – Admin, Developer2 und Developer5 – steht. Die Position von User9 leistet in diesen Zusammenhängen nach außen Repräsentations- und nach innen Koordinationsfunktion. Wie die Anzahl der Beziehungen innerhalb dieser Klasse und zum Zentrum deutet auch die Häufigkeit der Kommunikation auf eine Formation eigenständiger Qualität hin, die sich von den übrigen Usern signifikant unterscheidet.

Für die Positionen der Entwickler Admin, Developer2 und Developer5 sind insbesondere deren strukturelle Qualitäten als Gatekeeper/Representant und Liaison Broker von eminenter Bedeutung, die unter den verschiedenen Einteilungspartitionen variierend auftreten. Die Ergebnisse verdeutlichen die doppelte Funktion dieser Positionen, die zum einen als ‚Kommunikationspartner' für die User fungieren und zum anderen die ‚Übermittlung' dieser Informationen in den Kreis der Softwareentwickler realisieren und selbst aktiv zum Quellcode beitragen. Ihre primäre Leistung liegt damit in der Verbindung der Klassen.

Zusammenfassend lässt sich festhalten, dass sich das Kommunikationsnetzwerk des freien/offenen Softwareprojekts „Password Safe" in einem ausgeprägten Zentrum-Peripherie-Verhältnis zwischen Entwicklern und aktiven Anwendern der Software ausdifferenziert. Das Zentrum dieser Struktur konstituiert sich dabei über drei prominente Brokerpositionen, welche als Teil der in sich geschlossenen Entwicklerclique die Beziehungen zur Peripherie und Semi-Peripherie realisieren und organisieren. In diesem Sinne lässt sich das Zentrum in zwei Bestanteile unterteilen, die man als ‚Grenzstellen' des Außenkontaktes und eine dahinterliegende ‚Subeinheit' erfassen kann. Peripherie und Semi-Peripherie repräsentieren die User, die im ersteren Fall

(fast) keine ‚internen' Beziehungen zueinander unterhalten und sich in ihrem Kontakt mit den Grenzpositionen des Zentrums durch einzelne oder nur geringfügig häufigere Kommunikationsepisoden auszeichnen. Im Falle der Semi-Peripherie sind diese Beziehungen ausgeprägter. Diese relativ kleine Formation zeigt nach innen eine recht kohäsive Schließung auf. Darin ähnelt sie zwar der ‚Subeinheit', ist jedoch an der direkten Entwicklung des Quellcodes nicht beteiligt und stellt insofern eine eigene, vom Zentrum zu trennende Klasse dar.

6 Fazit

Die Ergebnisse der Fallanalyse des freien/offenen Softwareprojekts „Password Safe“ ermöglichen in Zusammenschau einen grundlegenden Einblick in die Bedingungen und Formen der Konstitution und Reproduktion des organisationalen Zusammenhanges. Es zeigt sich, dass die Kernproblematik der sozialen Ordnungsbildung im Internet schon in Bezug auf den recht gut abgrenzbaren Bereich der kooperativen Softwareentwicklung ein vielschichtiges Phänomen ist, welches einen entsprechend breiten Zugang und Erklärungsansatz benötigt, um ein maßgebliches Verständnis dieses Phänomens zu erlangen. Vor diesem Hintergrund ermöglicht ein systemtheoretischer Rahmen es, aus einer funktionalen Perspektive nach den Strukturen und Systemleistungen der Abgrenzung sozialer Systeme von einer Umwelt sowie der Gestaltung der System-Umwelt-Beziehungen zu fragen. Für die Erfassung dieser Zusammenhänge bietet er sowohl die theoretischen Voraussetzungen als auch eine entsprechend ausgearbeitete Begrifflichkeit.

Projekte der freien/offenen Softwareentwicklung differenzieren sich in einer spezifischen Umwelt aus, die als solche über einen allgemeinen Wertekanon mit elaborierter Semantik den kontextualen Sinnhorizont für die Systemdifferenzierung selektiert. Diese für die einzelnen Projekte spezifische (Teilsystem-)Umwelt lässt sich am besten als eine Form sozialer (Protest-)Bewegung erfassen: Sie (re-)produziert sich in Abgrenzung zu wirtschaftlichen und rechtlichen Entwicklungen der proprietären Softwareproduktion und konstituiert sich in ihren Anfängen primär über geteilte Wertideale der Freiheit, Selbstbestimmung und sozialen Gerechtigkeit. In ihrem weiteren Verlauf prägen sich verschiedene Semantiklinien aus, die diesen Wertekanon mit Vorstellungen über die geeigneten und zulässigen Mittel zur Verwirklichung der Werte verbinden und in Form einer Zweck-Mittel-Relation zunehmend erweitern. Dabei treten vor allem Aspekte der universalen Inklusion und Kooperation hervor, die sich unter dem symbolischen Medium ‚freie/offene Softwareentwicklung‘ aggregieren lassen und als solches die Annahme der Sinnprämissen (Kommunikations*erfolg*) sichern.

Projekte der freien/offenen Softwareentwicklung differenzieren sich unter diesen Rahmenstrukturen als organisationaler Kooperationszusammenhang „in einer komplexen und veränderlichen Umwelt durch Stabilisierung einer Innen/Außen-Differenz“ (Luhmann 1999b: 175) aus. Als soziales System grenzen sie sich durch Selektion von (Anschluss-)Möglichkeiten von ihrer Umwelt ab und reduzieren bzw. organisieren durch systememergente Relevanzstrukturen, die das Erleben und Handeln orientieren, Komplexität (Luhmann 1987).

> „Systembildung besteht mithin in der Stabilisierung relativ invarianter und auf die Umwelt bezogener Sinnstrukturen, die Komplexität reduzieren oder doch die Re-

duktion von Komplexität durch konkretes Verhalten erleichtern können“ (Luhmann 1999b: 187).

Damit wird für die systemtheoretische Analyse eines freien/offenen Softwareprojekts mithin die Bezugsproblematik deutlich, die in der Regulierung der System/Umwelt-Grenze(n) durch Prozesse und Strukturen des Systems im Medium des Sinns liegt. Der Fokus der Untersuchung des Projekts richtet sich damit auf die spezifischen Formen der Systemreproduktion und deren Erklärung vor dem Hintergrund einer emergenten Sinnordnung, die sich in diesen Formen manifestiert und getragen wird.

Im Falle des hier behandelten Zusammenhanges lassen sich eine Reihe unterschiedlicher, funktional äquivalenter Systemstrategien unterscheiden, die der Reduktion von Komplexität im Sinne einer (strukturellen) Ordnungsleistung dienen (Luhmann 1999b). Die wohl grundlegendste Form besteht im (generalisierten) Konsens der subjektiv gebildeten Vorstellungen im Bereich der freien/offenen Softwareentwicklung, der sich in einer in hohem Maße ausgeprägten Selbstverpflichtung auf die Gebote und Normen dieser Sinnwelt zeigt. Die geteilten und institutionalisierten Formen der Erlebnisverarbeitung (Wirklichkeitsdeutungen, Werte etc.) reduzieren so die Unendlichkeit an möglichen Verhaltensweisen und sichern ein gewisses Maß an Komplementarität des Erwartens (Luhmann 1999b). Dies ist gerade vor dem Hintergrund sehr geringer Sanktionsmöglichkeiten und des Erfordernisses eines hohen Maßes an Bereitschaft zum Engagement, welches sich nicht durch Entlohnung sichern lässt, für die Einheit und Stabilität des Systems von hoher Bedeutung. Insbesondere für die beteiligten Softwareentwickler dürfte das Identifikationspotenzial der freien/offenen Softwareentwicklung einen wichtigen Aspekt darstellen.

Auf der Ebene der Projektorganisation und -kommunikation lassen sich zwei weitere Strategien identifizieren: Umweltdifferenzierung und Binnendifferenzierung des Systems. Im ersteren Fall unterscheidet das System in der sozialen Dimension ganz elementar zwischen Entwicklern und (aktiven, gehobenen) Anwendern[46]. Diese Trennung zeigt sich sowohl im formalen Status der einzelnen Akteure auf der Plattform als auch in der inhärenten Gebrauchslogik der Organisationstools (s. o.), die zur Kommunikation zwischen diesen Gruppen angelegt sind und benutzt werden. Die Organisationstools differenzieren zudem in erheblichem Maße auf sachlicher Ebene und selektieren sowohl thematische Ausschnitte (Softwarefehler, gewünschte Leistungsmerkmale und Unterstützungsgesuche) als auch Formen der Kommunikation (sachlich strukturierte Eingabemaske und offener Diskussionsbeitrag). Mit ihrer Hilfe organisiert das System Unterscheidungen und Selektionen:

[46] Hier ließe sich ggf. unter Einbezug weiterer Dimensionen noch stärker differenzieren nach dem Grad des Informatikverständnisses und der inhaltlichen Beteiligung. Der Block der Semi-Peripherie zeigt z. B. deutlich höhere Beteiligungsraten sowie einen ausgeprägten Schwerpunkt im Bereich des Feature Request und ließe sich auf dieser Grundlage als Umwelt eigener Qualität unterscheiden.

> „Es bildet zu jeweils verschiedenen Umweltausschnitten je besondere Grenzen, stabilisiert an diesen Grenzen je besondere Beziehungen und gründet seine Autonomie und seine Fähigkeit zu Indifferenz gegenüber Umweltveränderungen gerade auf diese Unterschiedlichkeiten. […] Auf diese Weise kann das System sich spezialisieren, es sich also leisten, gegenüber einer Vielzahl von Umweltveränderungen indifferent sein. Auch gesicherte Indifferenz ist eine Art der Absorption von Komplexität und Veränderlichkeit“ (Luhmann 1999b: 184).

An diese Formen der Umweltdifferenzierung lassen sich Binnendifferenzierungen und Prozessdifferenzierungen anschließen, die sich als dritte Strategie empirisch vornehmlich in den Kommunikationsstrukturen des Projekts finden. Dies betrifft im betrachteten Fall vor allem die Strukturausprägungen der Clique der Entwickler, die als zentrale Formation ein Kern- bzw. Subsystem eigener Güte darstellen[47]. Die Ergebnisse der Struktur- und Positionsanalysen des Kommunikationsnetzwerkes zeigen hier eine grundlegende Differenzierung innerhalb dieser Clique: einerseits diejenigen Entwickler, die ohne jeden ‚Außenkontakt‘ am Quellcode der Software arbeiten, und andererseits solche Entwickler, welche ausgeprägte Brokerpositionen übernehmen und als Anschlusskommunikatoren für die Mitteilungen der Anwender fungieren. Diese spezialisieren sich, soweit zu erkennen, zwar nicht auf sachliche Bereiche – z. B. durch Zuständigkeiten für einzelne Tools –, realisieren aber als Gatekeeper zum Bereich des Quellcodes die Möglichkeit der Interdependenzunterbrechung.

> „Störende Umwelteinwirkungen können auf diese Weise in Teilen des Systems lokalisiert und abgekapselt werden; sie übertragen sich nicht ohne weiteres auf andere Teile, also nicht auf das Ganze, weil es infolge partieller Unabhängigkeit der Teile voneinander nur Effektübertragungen gibt, die funktional sinnvoll sind […] so ermöglicht die interne Differenzierung außerdem eine Anpassungsbeschleunigung (weil nicht jeweils das ganze System geändert werden muß), eine Steigerung der Lernfähigkeit durch Spezifikation und die Dauererhaltung von gelernter Anpassung für mögliche Fälle in Teilsystemen, die sich wechselseitig nicht behindern, und allgemein eine Entlastung der Teilsysteme von übermäßiger Komplexität“ (Luhmann 1999b: 185).

Die simultane Durchsetzung dieser verschiedenen Strategien lässt sich für ein soziales System subsumiert in Form einer Zwecksetzung realisieren, die als selektive Auswahl aus (konstruierten) ‚kausalen‘ Ursache-Wirkungs-Ketten „dem Grundproblem der Bestandserhaltung in einer komplexen und veränderlichen Umwelt, das als

[47] Die Begrifflichkeit folgt der graduellen Systemkonzeption des Theorieansatzes von Fuchs (2005), die der Luhmann’schen Typologie von Interaktion, Organisation und Gesellschaft nicht folgt. Letztere hier anzuwenden, bringt meiner Einschätzung nach keinen wesentlichen Erkenntnisgewinn; zudem arbeitet Luhmann im Rahmen der Organisation selbst mit dem Konzept der Clique (Luhmann 1999a). Eine Übertragung auf den Gegenstand würde am ehesten einer Klassifikation der Entwickler als Organisation und einer Behandlung der Kommunikation zwischen Anwendern und Entwicklern als „Interaktion“ oder „freie Kommunikationen“ entsprechen.

solches nicht instruktiv, nicht entscheidungsfähig ist, eine systemintern bearbeitbare Fassung“ (Luhmann 1999b: 190) gibt. Auf dieser Grundlage lassen sich Relationen von zu erstrebenden Wirkungen und geeigneten Mittel gewinnen, welche den Strukturaufbau und die Prozesse des Systems orientieren. Hierin liegt die Funktion der Zwecksetzung als „koordinierende Generalisierung“ (Luhmann 1999b: 189), die im Sinne einer abstrahierenden Relevanzordnung die Auslegung der ‚Wirklichkeit‘ durch das System leitet und über den Einzelfall hinaus stabilisiert und koordiniert.

Die Zwecksetzung dient dem System dann als Ersatzformel für sein Bestandsproblem, wenn es seine Wertinteressen durch relativ konstante Zwecksetzung generalisiert und eine zumindest rudimentäre Permanenz stabilisiert. Dieser Aspekt lässt sich auch als Invarianz der System/Umwelt-Grenze beschreiben, d. h. als Konstitution einer Sinnordnung, die aus den zahllosen Aspekten der Umwelt selektiert und diese mit einer spezifischen Relevanz auszeichnet (Luhmann, 1987). Eine solche Stabilisierung durch Differenzierung lässt sich jedoch nur in dem Maße realisieren, als der Systemzweck Anerkennung in der Umwelt findet und das System mit Leistungen vergütet, welche es zur Lösung seiner Systemprobleme einsetzten kann. Diese „generalisierten Medien der Problemlösung“ (Luhmann 1999b: 204) erhöhen die Freiheitsgrade des Systems umso mehr, je unspezifischer ihr Einsatz möglich ist, je mehr sie zeitlich, sachlich und sozial generalisiert sind.

Im Bereich der freien/offenen Softwareentwicklung bzw. im operationalen Zusammenhang der Softwareprojekte lässt sich, wie für freiwillige Vereinigungen typisch, als kritisches Medium das persönliche Engagement ausmachen. Damit gerät „die generalisierte Persönlichkeitsstruktur in den Blick, die den Menschen motivieren und bewegen kann, um bestimmter […] Wirkungen willen erhebliche Verhaltenslasten auf sich zu nehmen, weil ihm das Freude macht“ (Luhmann 1999b: 208). Eine solche gefühlsbasierte Leistung ist für soziale Systeme nicht ohne weiteres zu manipulieren und zeigt eine direkte Bindung an spezifische Sinngehalte, die in der Zwecksetzung zu berücksichtigen sind. Das betrifft zum einen die sachlich-inhaltliche Auswahl, aber auch den Grad an Bestimmtheit der Zweckformulierung (Luhmann 1999b). Für Letztere lassen sich zwei entgegensetzte, funktional äquivalente Weisen angeben: „durch Abstraktion auf eine spezifische Wirkung mit Indifferenz gegen andere Folgen des Handels oder durch inhaltliche Generalisierung zu einer unbestimmt gelassenen Wunschvorstellung, die gänzlich offen läßt, ob und welche Wege zum Ziel führen“ (Luhmann 1999b: 212). Diese Extrempunkte sind zuallererst theoretischer Natur, und Mischformen sind möglich und wohl eher der Regelfall.

Für den sozialen Zusammenhang der freien/offenen Softwareentwicklung liegt in diesem Aspekt ein bedeutender und hoher Erklärungswert. Die inhaltliche Ausrichtung auf das Thema Software und deren Entwicklung wählt zunächst einmal einen relativ

engen Sachverhalt als Kommunikationsgrundlage und begrenzt sich in diesem Sinne auf einen spezifischen Ausschnitt der gesellschaftlichen Umwelt. Innerhalb dieses Bereichs vollzieht sich dann jedoch eine erhebliche Ausweitung der sachlichen und sozialen Relevanzstrukturen, die durch Bezugnahme auf verschiedenste Themenvorräte aus dem Leitaspekt „freie/offene Software“ entwickelt werden können: Der Aufbau organisierter Komplexität setzt die Reduktion von Komplexität durch Selektion voraus (Luhmann 1987).

Aus der Perspektive eines Zweck-Mittel-Schemas ist in den Anfängen der Bewegung die Schwerpunktsetzung auf recht abstrakte Wertideale wie Freiheit von Wissen und Erkenntnis, Selbstbestimmung und soziale Gerechtigkeit auszumachen. Der Diskurs ist geprägt durch eine wertgeladene Semantik, für die ein emotionales Zusammengehörigkeitsgefühl von grundlegender Bedeutung sein dürfte. Im weiteren Verlauf werden dann zunehmend spezifische Mittel-Wirkungs-Ketten konstruiert, die vor allem die operative Realisierung durch Kooperation und Anspruch auf Inklusion hervorheben. In diesem Zuge kommt es zu einer erheblichen Anreicherung der Zwecksetzung, die sich sowohl auf die Ebene der gewünschten Wirkungen – z. B. das im Rahmen der Linux-Entwicklung entstehende Ideal technischer Exzellenz – als auch und insbesondere auf die Ebene der Mittel erstreckt. Wie an entsprechender Stelle ausgeführt, etablieren sich auf diesem Wege unterschiedliche, aber nebeneinander fortbestehende Semantiken, die insgesamt eine relativ heterogene Zwecksetzung repräsentieren, welche eine Vielzahl unterschiedlicher, teils widersprüchlicher Mittel-Wirkungs-Relationen selektiert und vereint.

Diese Vielfalt in der thematischen Breite (unter dem Leitaspekt Software) und die Variationen des Bestimmtheitsgrades der Zwecksetzung dürften wesentliche Bedingungen für den ‚Erfolg‘ der freien/offenen Softwareentwicklung sein: Die Selektion eines spezifischen thematischen Ausschnittes ermöglicht den systeminternen Aufbau einer Sinnordnung, die in Form der Zwecksetzung einen Komplex an Relevanzstrukturen orientiert und stabilisiert. Die Vielschichtigkeit der Zweckformulierung bietet dabei Möglichkeiten des Anschlusses verschiedener Umwelten, die im vorliegenden Fall für die Sicherung von Engagement – als generalisiertes Problemlösungsmedium – unerlässlich ist. Zum Beispiel können Entwickler ebenso aus der Überzeugung der technischen Qualität freier/offener Software wie auch aus Vorstellungen sozialer Gerechtigkeit heraus als Mitglieder gewonnen werden, genauso wie Anwender sich mit diesen Idealen identifizieren können oder vielleicht nur die personalisierte, kostenlose Software schätzen. Die abstrahierende Zwecksetzung leistet dennoch einen grundlegenden ‚Konsens‘ über die notwendigen Mittel – siehe obige Kontextanalyse –, die in Form entsprechender Sinnprämissen und Erwartungsstrukturen dem System eine Ordnung geben (Luhmann 1999b).

> „Die Variation des Grades an Zweckspezifikation ist die Art und Weise, in der die verschiedenen Funktionsrichtungen der Zwecksetzung kombiniert werden können. Nur weil das Zweckprinzip in diesem Sinne elastisch ist, kann eine Zwecksetzung als ‚subjektiv' erlebt werden, kann sie Umweltinstitutionen und Umweltdifferenzierungen angepasst und zugleich intern hinreichend unbestimmt und doch instruktiv und differenzierbar formuliert werden" (Luhmann 1999b: 214).

Der Strukturierungswert der Zwecksetzung findet sich auch auf der Ebene der Projektorganisation, wie sie oben als einzelne Strategien dargestellt wurden. Am deutlichsten zeigt sich dies in den sozialen und sachlichen (Umwelt-)Differenzierungen sowie den entsprechenden Grenzstellen, die unter dem Gesichtspunkt der Inklusion und Kooperation organisiert sind. Die Funktion dieser Strukturen wird jedoch erst gänzlich einsichtig, wenn man auf die Prozesslichkeit des Kommunikationszusammenhanges abstellt und dazu auf ein Input/Output-Modell zurückgreift (Luhmann 1999b).

> „Das Input/Output-Modell [hat] die zunächst sehr einfache Form der Vorstellung eines Kommunikationsflusses, der durch Schwellen markiert ist, die der Innen/Außen-Differenz von Systemen entsprechen. Der Kommunikationsfluss bringt an einer oder mehreren Stellen Informationen von außen in das System, dort werden sie verarbeitet, kombiniert und umgestaltet, gefiltert und verdichtet, und verlassen dann an anderen Stellen als Kommunikationen oder Entscheidungen das System" (Luhmann 1999: 249f.).

Aus einer solchen Perspektive kann das System – insbesondere Information verarbeitende Systeme – Mittel als Input und Zweck als Output betrachten und ein Raster für die Suche und Auswahl an Information, die es in Bezug auf seinen Output für die Kommunikation braucht, entwickeln. Nimmt es dabei zu mehreren, verschiedenen Umwelten Beziehungen auf, d. h. unter den Bedingungen der Umweltdifferenzierung, lassen sich Input und Output als Systemgrenzen differenzieren und die Organisations- und Programmstruktur entsprechend dieser Unterscheidungen einrichten (Luhmann 1999b).

> „Organisationen, ja Systembildung überhaupt, kann in ihrem Kern als stets prekäre Übersetzung von Funktionen in Strukturen begriffen werden. Innerhalb von Systemen sind Funktionen Beziehungen von Leistungen auf diejenigen Systemprobleme, die gelöst werden müssen, damit ein System bestehen kann. Da diese Probleme Dauerprobleme sind, muss in dauerhafter Weise für sie gesorgt werden. Das kann auf rationale Weise vor allem dadurch geschehen, daß das System seine Struktur auf seine Probleme ausrichtet, indem es seine Probleme programmiert, die erforderlichen Verhaltensweisen normiert und sie zu funktionsspezifischen Untersystemen zusammenführt" (Luhmann 1999b: 261).

Anhand der Kommunikationsstrukturen des Projekts „Password Safe“ sowie der virtuellen Organisationstools lassen sich die Konstitution von Subsystemen anhand von Umweltgrenzen und die Beziehung dieser Teileinheiten zueinander beobachten. Die Produktion der eigentlichen Software vollzieht sich in einem relativ abgeschlossenen kommunikativen Zusammenhang durch die tragende Umwelt der Entwickler. Die engen und direkten Kommunikationsbeziehungen der relativ kleinen Anzahl an Mitgliedern deuten auf eine primär konsensorientierte Entscheidungsfindung hin; bei genauerer Betrachtung des CVS-Tools zeigen sich allerdings Hinweise auf eine informelle Hierarchisierung, die sich im Ausmaß der Eingriffe in den Quellcode ausdrückt[48]. Der Besitz von Schreibrechten als Mitgliedschaftsbedingung verdeutlicht die Schließung dieser Subeinheit[49], der in diesem Sinne ein gewisses Maß an Formalität zu attestieren ist. Auf sozialer Ebene lassen sich als zweiter wesentlicher Umweltausschnitt die aktiven Anwender der Software zusammenfassen, die sich als solche empirisch zunächst einmal durch ihre Registrierung auf der Plattform als potenzielle Umwelt auszeichnen und deren Beiträge in den verschiedenen Organisationstools der Projekt- bzw. Systemkommunikation zugeordnet werden können. Diese Umwelt wird mittels der Organisationstools vor allem auf sachlicher Dimension erheblich weiter differenziert (s. o.). Aufgrund des sachlichen Primats, welches noch verstärkt wird durch die Schwerpunktsetzung internetbasierter Kommunikation auf die Information, lassen sich die entsprechenden Kommunikationen dem Differenzierungsprinzip der Tools folgend ebenso als Subeinheiten abgrenzen.

In einer Innenperspektive des Systems fungieren die verschiedenen Untersysteme jeweils als Umwelt füreinander und stellen sich als solche gegenseitig verringernde Komplexität zur Verfügung. Die Beziehungsstruktur der Einheiten untereinander folgt einem Zentrum-Peripherie-Verhältnis, wobei der Kern des Systems in der Quellcodeproduktion liegt. Das bedeutet: Der Output der übrigen Organisationseinheiten stellt den Input für die Subeinheit der Quellcodeproduktion dar, die in den verschiedenen, verbesserten Softwarereleases ihren Output finden. Die Leistung dieser Verbindungen liegt also in der Verarbeitung spezifischer Informationen, die sachlich geordnet seitens der Subsysteme zur Verfügung gestellt werden. Letztere wiederum gewinnen ihren Output aus der Anwendung der Software. Den virtuellen Kommunikationsräumen (Tools) kommt in ihrer Funktion als „Vermittler“ daher eine elementare Bedeutung zu, die im Sinne von ‚Grenzstellen‘ die Reduktion von Komplexität realisieren (Luhmann 1999a). Im Falle der Tracker, die explizit auf die Transformation ‚einzelner‘ Informationen in der Software ausgelegt sind, strukturieren sie durch enge Formvor-

[48] Dieses Tool wurde nicht explizit analysiert, jedoch fielen im Zuge der Erhebung der Kommunikationsbeiträge zu diesem Tool drei grundlegend differenzierbare Formen der Beteiligung auf, die sich auch unterschiedlich auf die Akteure verteilten: der einfache Beitrag zum Quellcode (1), die Veränderung des bestehenden Quellcodes (2) und die Löschung von Quellcode (3).
[49] Zu den (funktionssichernden) Hürden des Eintritts in diese Gruppe vgl. Fußnote 14.

gaben die Kommunikation. In diesen (sachlichen) Schranken prozessieren sie eine konditionale Programmierung: Die Abgabe von Mitteilungen durch den Anwender fungieren als Auslöser des Handelns für die Entwickler. Das System ermöglicht so eine *von außen* auslösbare Einflussnahme und stellt eine allgemeine Reaktionsbereitschaft in Aussicht (Luhmann 1999a, 1999b). Im Falle der Foren sind diese Vorgaben deutlich geringer, sowohl in Bezug auf die Form als auch hinsichtlich der sachlichen Dimension. Allerdings ist hier auch der unmittelbare Zusammenhang zur Entwicklung des Quellcodes nicht so stark gegeben und die Ereignisse in diesen Tools bedürfen im jeweiligen Einzelfall noch stärker einer systememergenten Schematisierung. Der Gewinn liegt dabei in einer prinzipiellen Offenheit für ‚unvorhersehbare' Umweltkonstellationen (Luhmann 1999b).

Die Kommunikationsstrukturen des Projekts zeigen entsprechend idealtypisch über die Tools vermittelte Kommunikationsbeziehungen zwischen Anwendern und Entwicklern. Die relativ kurzen Kommunikationsepisoden dürften ein Hinweis auf das Leistungsvermögen der Tools durch Vorstrukturierung der Informationen sein, allerdings lassen sich auch keine signifikanten Unterschiede zwischen den Tracker-Systemen und den Foren ausmachen. Um diesen Aspekt belastbar zu beurteilen, müssten zudem die ‚Komplexitätsanforderungen' der jeweiligen Bereiche abschätzbar sein. Die überdurchschnittliche Beteiligung einiger weniger Akteure in der Sektion Feature Request, welche sich zu großen Teilen mit der Semi-Peripherie überschneiden, eröffnen Spekulationsmöglichkeiten, ob das (konzeptionelle) Entwickeln neuer, potenzieller Zusatzfunktionen der Software nicht vielleicht schwieriger ist als die Meldung von Softwarefehlern und daher entsprechender Kompensation bedarf – in diesem Fall durch beständige Beteiligung und wiederholten Austausch einer begrenzter Anzahl an Akteuren.

Neben den Organisationstools lassen sich den Broker-Positionen wichtige Selektionsleistungen im organisationalen Kommunikationszusammenhang zuschreiben. An Grenzstellen der Entwicklergruppe sichern sie für sachlich *alle* Bereiche *alleinig* den Anschluss an die Anwender-Umwelten und entlasten so „ihre" Einheit. Diese Leistung wird vor dem Hintergrund alternativer Strategie deutlich, z. B. der Einbindung aller Entwickler in die Kommunikation mit den Anwendern, dann eventuell nach zeitlichen oder sachlichen Kriterien strukturiert. Ihre Selektionsfunktion zeigt sich aber auch anhand der ‚separierten' Kommunikationsmenge des Netzwerkes, die sich um das Thema einer „Portable-App" (Komponente 11) konstituiert: Die Entwickler knüpfen an diesen Möglichkeitsraum nicht an und schließen ihn insofern als ‚nicht relevante' Sinneinheit vom System aus. Die Redundanz der Kommunikationsbeziehungen attestiert in diesem Sinne sowohl eine gewisse operative Routine als auch eine gewisse Rolleninstitutionalisierung, die den organisationalen Zusammenhang durch

Erwartungsstrukturen stabilisieren und die Prozessabläufe, besonders ‚zwischen' den Subsystemen, ganz wesentlich tragen und selegieren (Luhmann, 1999a).

Der Binnendifferenzierung entsprechend können bzw. müssen sich auch die Zweckperspektiven der Einheiten verschieben. Für den untersuchten Gegenstand zeigt dieser Aspekt dabei eine besondere Form: Einerseits konstituiert sich das System über einen sehr spezifischen Gegenstand, die Software „Password Safe", und findet entsprechend Anschluss an eine stark begrenzte Umwelt, die Anwender und Entwickler der Software. Input und Output beziehen sich damit ‚gleichermaßen' auf strukturell ähnliche Umwelten, so dass strukturelle Erfordernisse zur Systemdifferenzierung nur begrenzt vorhanden sind. Auf der anderen Seite ist diese Differenz nicht von der Hand zu weisen, und es schließen sich, wie gezeigt, charakteristische Systemdifferenzierungen an sie an. Legt man vor diesem Hintergrund als maßgeblichen Unterschied der sozialen Einheiten die Fähigkeit zur Programmierung von Software zugrunde und stellt die hohe Anzahl einmalig bzw. sporadisch beitragender Anwender in Rechnung, eröffnet sich ein umfassenderes Bild: Während sich die Mitteilungen seitens der Anwender aus einem Aggregat persönlicher Erfahrungen, Probleme und Anliegen zusammensetzen, die diese mittels der Entwickler in die Software einbringen wollen, dienen den Entwicklern die Mitteilungen als Feedback und Informationsquelle zur Verbesserung und Entwicklung der Software. Für beide fungiert der jeweils andere als Mittel zum Zweck. Das heißt nicht, dass Entwickler und Anwender sich unpersönlich oder gar opportunistisch gegenüberstehen – das muss nicht so sein, kann aber. Der entscheidende Punkt – und erst vor diesem Hintergrund ist ein funktionales Verständnis der spezifischen Strukturausprägungen des Projekts möglich – liegt in der *sozialen Ordnungsleistung* der Zwecksetzung bzw. -konstruktion freier/offener Softwareentwicklung: als *emergentes* Relevanzsystem orientiert es sinnhaft das Verhalten der Akteure und ermöglicht die soziale Kooperation einer Vielzahl potenziell anonymer und heterogener Akteure im Internet.

> „Eine ausreichende Koordination […] kann nun auf sehr verschiedene Weisen zustandekommen. Sie braucht nicht über eine Hierarchie oder über einen gemeinsamen Zweck zu laufen. Sie setzt nicht einmal eine ausdrückliche Koordinationsentscheidung voraus. Wenn die Untersysteme aber in einem selbst zweckspezifisch strukturierten Gesamtsystem zusammenarbeiten sollen […], muß für eine bestimmte Ordnung des Zusammenwirkens gesorgt werden. Es muß dann ein Zweck/Mittel-Gefüge höherer Ordnung entworfen werden, das in den Output des Gesamtsystems mündet und mit den Einzelzwecken der Untersysteme als Mittel rechnet. Dieser Ordnung wird durch Zweckprogrammierung erstellt und verbindlich gemacht" (Luhmann 1999b: 273).

7 Resümee

Der in dieser Arbeit gewählte Forschungsansatz ermöglicht anhand der Fallanalyse eines Projekts freier/offener Softwareentwicklung weitreichende Einsichten in die Formen sozialer Ordnungsbildung internetbasierter Kooperationszusammenhänge. Mittels der systemtheoretischen Perspektive lassen sich die Strukturen und Prozesse der (Re-)Produktion sozialer Systeme als Ausdruck emergenter Relevanzstrukturen erfassen und unter dem Aspekt der Funktionalität einer differenzierten Betrachtung zuführen. Das Erkenntnispotenzial einer solchen Herangehensweise lässt sich verdichten, insoweit man es dem Vergleich zuführt und die Beobachtungen in ihrer Qualität des ‚Auch-anders-möglich-Seins' herausarbeitet (Luhmann 1974a).

Im Rahmen einer exemplarischen Fallanalyse ist dies nur begrenzt möglich. Dabei lassen sich zwar in Rückgriff auf theoretische Überlegungen – der systemtheoretische Rahmen bietet hier elaborierte Grundlagen – eine Reihe ergiebiger Anschlussmöglichkeiten gewinnen, die jedoch die Erfassung der empirischen Realität in ihrer Variationsbreite nicht zu ersetzen vermag. In diesem Sinne ließe sich die Untersuchung in einer Analyse von weiteren Projekten der freien/offenen Softwareentwicklung oder auch anderer Kooperationszusammenhänge im Internet weiterführen und einer vergleichenden Perspektive zuführen. Für den vorliegenden Bereich der freien/offenen Softwareentwicklung gilt dies in besonderem Maße, da sich eine Vielzahl unterschiedlicher Strukturgebilde zeigen, die von Verbänden mehrerer kleiner, einzelner Projekte bis hin zu sehr großen, in sich geschlossenen Organisierungen reichen. Insgesamt erreichen im Bereich der freien/offenen Softwareentwicklung nur wenige Projekte einen langfristigen und nachhaltig produktiven Zustand, von diesen zeigt das hier ausgewählte Projekt jedoch einen relativ weitverbreiteten und in diesem Sinne typischen Charakter bezüglich Größe und Kommunikationsaktivität (Krishnamurthy 2005). Es ließen sich jedoch zudem Unterschiede in der Komplexität des Arbeitsgegenstandes (Softwareart, -ziel etc.) als auch in den relevanten Umwelten (private Anwender, Firmen, Forschungsinstitutionen etc.) unterscheiden. Eine vergleichende Analyse könnte hier weitere Zusammenhänge aufzeigen und das Verhältnis verschiedener Systemausprägungen und Sinnordnungen ausarbeiten. Daran ließen sich dann auch praktische Fragen, z. B. unter dem Aspekt des Erfolges, anschließen.

Ein weiterer Gesichtspunkt betrifft das analytische Auflösevermögen, welches sich unter methodischer Erweiterung des Ansatzes erhöhen ließe und die Untersuchung struktureller Systemdifferenzierungen um eine Ebene operativer Sinnprozesse ergänzen könnte. Damit ist vor allem ein hermeneutischer Zugang zu der Projektkommunikation gemeint, der über die Rekonstruktion latenter Sinnstrukturen, ggf. auch zu Aspekten der Konfliktbewältigung oder Konsensherstellung, wichtige Beiträge zu einem umfassenden Verständnis der Systemorganisation beitragen könnte. Die Arte-

faktanalyse leistet in diesem Rahmen einen ersten Beitrag, der wesentliche Einsichten in sinnkonstituierte Strukturprinzipien der Projektkommunikation gibt. Diese Perspektive ließe sich jedoch noch erweitern und auf relevante Ausschnitte, wie z. B. die Kommunikation zwischen den Entwicklern, übertragen. Ein solches Vorgehen wäre allerdings mit der speziellen Sprachlichkeit der Kommunikation im Zusammenhang von (freier/offener) Softwareentwicklung konfrontiert und wäre wahrscheinlich nur in interdisziplinärer Zusammenarbeit realisierbar. Dann würden sich jedoch weitreichende Untersuchungsmöglichkeiten ergeben, die unter der obigen Anordnung des Vergleichens weitere Fragestellungen und die Grundlage für umfassende Abstraktionen eröffnen.

8 Literaturverzeichnis

Ahrens, Daniela, 2003: Die Ausbildung hybrider Raumstrukturen am Beispiel technosozialer Zusatzräume. In: Funken, Christiane; Löw, Martina (Hrsg.), Raum – Zeit – Medialität. Interdisziplinäre Studien zu neuen Kommunikationstechnologien. Opladen: Leske und Budrich, 173–190.

Allmendinger, Jutta (Hg.), 2001: Gute Gesellschaft? Konstruktion sozialer Ordnungen. Verhandlungen des 30. Kongresses der Deutschen Gesellschaft für Soziologie in Köln. Opladen: Leske und Budrich.

Apelt, Maja; Tacke, Veronika, 2012: Einleitung. In: Apelt, Maja; Tacke, Veronika (Hg.), Handbuch der Organisationstypen. 1. Auflage, Wiesbaden: Springer VS für Sozialwissenschaften, 7–20.

Apelt, Maja; Tacke, Veronika (Hg.), 2012: Handbuch der Organisationstypen. 1. Auflage, Wiesbaden: Springer VS.

Baerisch, Stefan, 2005: Versionskontrollsysteme in der Softwareentwicklung. GESIS, Informationszentrum Sozialwissenschaften. IZ-Arbeitsbericht Nr. 36. http://www.gesis.org/fileadmin/upload/forschung/publikationen/gesis_reihen/iz_arbeitsberichte/ab_36.pdf, 04.02.2012.

Beck, Klaus, 2003: No sense of place? Das Internet und der Wandel von Kommunikationsräumen. In: Funken, Christiane; Löw, Martina (Hrsg.), Raum – Zeit – Medialität. Interdisziplinäre Studien zu neuen Kommunikationstechnologien. Opladen: Leske & Budrich, 119–138.

Beck, Klaus, 2006: Computervermittelte Kommunikation im Internet. Oldenbourg: Wissenschaftsverlag.

Braun-Thürmann, Holger, 2006: Ethnografische Perspektiven: Technische Artefakte in ihrer symbolisch-kommunikativen und praktisch-materiellen Dimension. In: Rammert, Werner; Schubert, Cornelius (Hg.), Technografie. Zur Mikrosoziologie der Technik. Frankfurt am Main: Campus Verlag, 199–222.

Bretthauer, David, 2001: Open Source Software: A History. UConn Libraries Published Works, Paper 7. http://digitalcommons.uconn.edu/libr_pubs/7, 27.03.2012.

Bomme, Michael; Tacke, Veronika, 2007: Netzwerke in der ‚Gesellschaft der Gesellschaft'. Funktion und Folgen einer doppelten Begriffsverwendung. In: Baecker, Dirk; Gutter, Michael; Romano, Gaetano; Stichweh, Rudolf (Hg.), Zehn Jahre danach. Niklas Luhmanns ‚Die Gesellschaft der Gesellschaft'. Themenheft Soziale Systeme, 2007, Jg. 13, Heft 1/2, 9–21.

Borgatti, Steve; Everett, Martin G.; Freeman, Lin. C., 2002: UCINET for Windows: Software for Social Network Analysis. Harvard/MA: Analytic Technologies.

Carillo, Kevin; Okoli, Chitu, 2008: The Open Source Movement: A Revolution in Software Development. The Journal of Computer Information Systems, 2008/2009, Vol. 49, No. 2, 1–9.

Crowston, Kevin; Howison, James, 2005: The social structure of free and open source software development. In: First Monday (Peer-Reviewed Journal of the Internet), 2005, Vol. 10, No. 2, 1–27. http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/rt/printerFriendly/1478/1393, 18.01.2012.

Crowston, Kevin; Howison, James; Annabi, Hala 2006: Information Systems Success in Free and Open Source Software Development: Theory and Measures. Software Process: Improvement and Practice, 2006, Vol. 11, Issue 2, 123–148.

De Nooy, Wouter; Mrvar, Andrej; Batagelj, Vladimir, 2011: Exploratory Social Network Analysis with Pajek. 2. Edition, Cambridge: Cambridge University Press.

DiBona, Chris; Ockman, Sam; Stone, Mark, 1999: Open Sources: Voices of the Open Source Revolution. Sebastopol: O'Reilly Media.

Döhring, Nicola, 2003: Sozialpsychologie des Internets. Die Bedeutung des Internets für Kommunikationsprozesse, Identitäten, soziale Beziehungen und Gruppen. 2., vollst. uberarb. und erw. Auflage, Göttingen: Hogrefe.

Ducheneaut, Nicolas, 2005: Socialization in an Open Source Software Community: A Socio-Technical Analysis. Computer Supported Cooperative Work, 2005, Vol. 14, No. 4, 323–368.

Elliott, Margaret S.; Scacchi, Walt, 2003: Free software developers as an occupational community: resolving conflicts and fostering collaboration. In: Proceedings of the 2003 international ACM SIGGROUP conference on supporting group work (GROUP '03), New York: ACM, 21–30.

Elliott, Margaret S.; Scacchi, Walt, 2004: Free Software Development: Cooperation and Conflict in a Virtual Organization Culture. In: Koch, Stefan (ed.), Free/Open Source Software Development. Hershey: Igi Global, 152–173.

Elliott, Margaret S.; Scacchi, Walt, 2008: Mobilization of Software developers: the free software movement. Informations Technology & People, 2008, Vol. 21, Issue 1, 2008, 4–33.

Ellrich, Lutz, 2002: Die Realität virtueller Räume. Soziologische Überlegungen zur „Verortung" des Cyberspace. In: Maresch, Rudolf; Werber, Niels (Hg.), Raum – Wissen – Macht. Frankfurt am Main: Suhrkamp, 92–113.

Ellrich, Matthias, 2004: Koordination und Kommunikation in Open-Source-Projekten. In: Gehring, Robert A.; Lutterbeck, Bernd (Hg.), Open Source Jahrbuch 2004: Zwischen Softwareentwicklung und Geschäftsmodell. Berlin: Lehmanns Media.

Esposito, Elena, 1993: Der Computer als Medium und Maschine. Zeitschrift für Soziologie, 1993, Jg. 22, Heft 5, 338–354.

Esposito, Elena, 1995: Illusion und Virtualität. Kommunikative Veränderungen der Fiktion. In: Rammert, Werner (Hg.), Soziologie und künstliche Intelligenz. Produkte und Probleme einer Hochtechnologie. Frankfurt am Main: Campus Verlag, 187–216.

Esposito, Elena, 2002: Virtualisierung und Divination. Formen der Räumlichkeit der Kommunikation. In: Maresch, Rudolf; Weber, Niels (Hg.), Raum – Wissen – Macht. Frankfurt am Main: Suhrkamp, 33–48.

Faßler, Manfred (Hg.), 1999: Alle möglichen Welten. Virtuelle Realität – Wahrnehmung – Ethik der Kommunikation. 1. Auflage, München: Wilhelm Fink Verlag.

Feller, Joseph; Fitzgerald, Brian, 2000: A framework analysis of the open source software development paradigm. In: Proceedings of the twenty first international conference on Information systems (ICIS '00). Association for Information Systems, Atlanta, 58–69.

Freyermuth, Gundolf S., 2007: Offene Geheimnisse – Die Ausbildung der Open-Source-Praxis im 20. Jahrhundert. In: Lutterbeck, Bernd; Bärwolff, Matthias; Gehring, Robert A. (Hg.), Open Source Jahrbuch 2007. Berlin: Lehmann Media.

Froschauer, Ulrike, 2009: Artefaktanalyse, In: Kühl, Stefan; Strodtholz, Petra; Taffertshofer, Andreas (Hg.), Handbuch Methoden der Organisationsforschung. Quantitative und Qualitative Methoden. Wiesbaden: VS Verlag für Sozialwissenschaften, 326–347.

Froschauer, Ulrike, Lueger, Manfred, 2007: Film-, Bild- und Artefaktanalyse. In: Straub, Jürgen; Weidemann, Arne; Weidemann, Doris (Hg.), Handbuch interkultureller Kommunikation und Kompetenz. Grundbegriffe – Theorien – Anwendungsfelder, Stuttgart: J. B. Metzler/Carl Ernst Poeschel Verlag, 428–439.

Fuchs, Peter, 1997: Adressabilität als Grundbegriff der soziologischen Systemtheorie. Soziale Systeme, Jg. 3, Heft 1, 57–79.

Fuchs, Stefan, 2005: Against Essentialism: A Theory of Culture and Society. First Paperback, Cambridge: Harvard University Press.

Fuhse, Jan A., 2005: Persönliche Netzwerke in der Systemtheorie. Schriftenreihe des Instituts für Sozialwissenschaften der Universität Stuttgart (SISS), No. 1.

Funken, Christiane; Löw, Martina (Hrsg.), 2003: Raum – Zeit – Medialität. Interdisziplinäre Studien zu neuen Kommunikationstechnologien. 1. Auflage, Opladen: Leske und Budrich.

Gehring, Robert A.; Lutterbeck, Bernd (Hg.), 2004: Open Source Jahrbuch 2004: Zwischen Softwareentwicklung und Geschäftsmodell. Berlin: Lehmann Media. http://www.opensourcejahrbuch.de/download/jb2004/OpenSourceJahrbuch2004.pdf, 01.07.2012.

Ghosh, Rishab A.; Glott, Ruediger; Krieger, Bernhard; Robles, Greogrio; 2002: Free/Libre and Open Source Software:survey and study. Deliverable D18: Final Report. Part IV: Survey of Developers. International Institute of Infonomics, University of Maastricht and Berlecon Research GmbH. http://www.math.unipd.it/~bellio/FLOSS%20Final%20Report%20-%20Part%204%20-% 20Survey%20of%20Developers.pdf, 01.10.2012.

Hanneman, Robert A.; Riddle, Mark, 2005: Introduction to social network methods. Riverside/CA: University of California. http://faculty.ucr.edu/~hanneman/, 28.09. 2012.

Hellmann, Kai-Uwe (Hg.), 1997: Protest: Systemtheorie und soziale Bewegungen. 2. Auflage, Frankfurt am Main: Suhrkamp.

Himaen, P., 2001. Die Hacker-Ethik und der Geist des Informations-Zeitalters. 1. Auflage, München: Riemann Verlag.

Hollstein, Betina, 2006: Qualitative Methoden und Netzwerkanalyse – ein Widerspruch? In: Hollstein, Betina; Straus, Florian (Hg.), Qualitative Netzwerkanalysen. Konzepte, Methoden, Anwendungen. Wiesbaden: VS Verlag für Sozialwissenschaften, 11–36.

Hollstein, Betina; Straus, Florian (Hg.) 2006: Qualitative Netzwerkanalysen. Konzepte, Methoden, Anwendungen. 1. Auflage, Wiesbaden: VS Verlag für Sozialwissenschaften.

Holtgrewe, Ursula, 2000: Kreativität als Norm – zum Erfolg verdammt? Open-Source-Software zwischen sozialer Bewegung und technischer Innovation. In: Allmendinger, Jutta (Hg.), Gute Gesellschaft? Konstruktion sozialer Ordnungen. Verhandlungen des 30. Kongresses der Deutschen Gesellschaft für Soziologie in Köln. Opladen: Leske und Budrich, 399–424.

Howison, James; Crowston, Kevin, 2004: The perils and pitfalls of mining SourceForge. Proc. of Workshop on Mining Software Repositories at the International Conference on Software Engineering {ICSE}. http://floss.syr.edu/publications/howison04msr.pdf, 28.11.2011.

Iivari, Netta, 2008: Empowering the users? A crictical textual analysis of the role of users in open source software development. AI & Society, 2008, Vol. 23, Issue 4, 511–528.

Jaeger, Till; Metzger, Axel, 2002: Open Source Software. Rechtliche Rahmenbedingungen der Freien Software. 1. Auflage, München: C. H. Beck Verlag.

Jahraus, Oliver (Hg.), 2001: Niklas Luhmann. Aufsätze und Reden. 1. Auflage, Stuttgart: Reclam Verlag.

Jansen, Dorothea, 2006: Einführung in die Netzwerkanalyse. Grundlagen, Methoden, Forschungsbeispiele. 3. Auflage, Wiesbaden: VS Verlag für Sozialwissenschaften.

Krishnamurthy, Sandeep, 2005: Cave or Community? An Empirical Examination of 100 Mature Open Source Projects. In: First Monday (Peer-Reviewed Journal of the Internet), 2005, Special Issue 2, http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/1477/1392, 18.01.2012.

Kerscher, Gottfried, 2000: Kopfräume – Eine kleine Zeitreise durch virtuelle Räume. 1. Auflage, Kiel: Ludwig.

Kling, Rob; Iacano, Suzanne, 1988: The Mobilization of Support for Computerization: The Role of Computerization Movements. Social Problems, Vol. 35, No. 3, 226–243.

Koch, Stefan (Hg.), 2004: Free/Open Source Software Development. Hershey: Igi Global.

Kühl, Stefan; Strodtholz, Petra; Taffertshofer, Andreas (Hg.), 2009: Handbuch Methoden der Organisationsforschung. Quantitative und Qualitative Methoden. 1. Auflage, Wiesbaden: VS Verlag.

Kuhm, Klaus, 2003: Telekommunikative Medien und Raumstrukturen der Kommunikation. In: Funken, Christiane; Löw, Martina (Hg.), Raum – Zeit – Medialität. Interdisziplinäre Studien zu neuen Kommunikationstechnologien. Opladen: Leske & Budrich, 97–118.

Latour, Bruno, 2010: Eine neue Soziologie für eine neue Gesellschaft: Einführung in die Akteur-Netzwerk-Theorie. Ins Deutsche übers. v. Gustav Roßler, 1. Auflage, Frankfurt am Main: Suhrkamp Verlag.

Lehmann, Franke, 2004: FLOSS Development as a social formation. First Monday (Peer-Reviewed Journal of the Internet), 2004, Vol. 9, Nr. 11. http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/1186/1106, 18.01.2012.

Lemley, Mark A., 1995: Convergence in the Law of Software Copyright? Berkeley Technology Law Journal, Vol. 10, Issue 1. http://www.btlj.org/data/articles/vol10/Lemley.pdf, 27.08.2012.

Ljungberg, Jan, 2000: Open source movements as a model for organizing. European Journal of Information Systems, 2000, Vol. 9, No. 4, 208–216.

Lueger, Manfred, 2010: Interpretative Sozialforschung: Die Methoden. 1. Auflage Wien: Facultas.wuv.

Luhmann, Niklas, 1972: Einfache Sozialsysteme. In: Zeitschrift für Soziologie, Jg. 1, Heft 1, 51–65.

Luhmann, Niklas, 1974a: Funktionale Methode und Systemtheorie. In: Luhmann, Niklas (Hg.), Soziologische Aufklärung 1. Aufsätze zur Theorie sozialer Systeme. 4. Auflage, Opladen: Westdeutscher Verlag, 31–53.

Luhmann, Niklas, 1974b: Funktion und Kausalität. In: Luhmann, Niklas (Hg.), Soziologische Aufklärung 1. Aufsätze zur Theorie sozialer Systeme. 4. Auflage, Opladen: Westdeutscher Verlag, 9–30.

Luhmann, Niklas, 1981a: Vorbemerkungen zu einer Theorie sozialer Systeme. In: Luhmann, Niklas (Hg.), Soziologische Aufklärung 3. Soziales System, Gesellschaft, Organisation. Opladen: Westdeutscher Verlag, 11–24.

Luhmann, Niklas, 1981b: Die Unwahrscheinlichkeit der Kommunikation. In: Luhmann, Niklas (Hg.), Soziologische Aufklärung 3. Soziales System, Gesellschaft, Organisation. Opladen: Westdeutscher Verlag, 25–34.

Luhmann, Niklas, 1986a: Systeme verstehen Systeme. In: Luhmann, Niklas; Schorr, Karl Eberhard (Hg.), Zwischen Intransparenz und Verstehen. Fragen an die Pädagogik. Frankfurt am Main: Suhrkamp, 72–117.

Luhmann, Niklas, 1986b: Alternative ohne Alternative. Die Paradoxie der ‚neuen sozialen Bewegungen'. In: Hellmann, Kai-Uwe (Hg.), Protest: Systemtheorie und soziale Bewegungen. Frankfurt am Main: Suhrkamp, 75–78.

Luhmann, Niklas; Schorr, Karl Eberhard (Hg.), 1986: Zwischen Intransparenz und Verstehen. Fragen an die Pädagogik. 1. Auflage, Frankfurt am Main: Suhrkamp Verlag.

Luhmann, Niklas, 1987: Soziale Systeme. Grundriß einer allgemeinen Theorie. 1. Auflage, Frankfurt am Main: Suhrkamp.

Luhmann, Niklas, 1990a: Haltlose Komplexität. In: Luhmann, Niklas (Hg.), Soziologische Aufklärung 5. Konstruktivistische Perspektiven. Opladen: Westdeutscher Verlag, 59–76.

Luhmann, Niklas, 1990b: Das Erkenntnisprogramm des Konstruktivismus und die unbekannt bleibende Realität. In: Luhmann, Niklas (Hg.), Soziologische Aufklärung 5. Konstruktivistische Perspektiven. Opladen: Westdeutscher Verlag, 14–58.

Luhmann, Niklas, 1994: Systemtheorie und Protestbewegungen. Ein Interview. In: Hellmann, Kai-Uwe (Hg.), Protest: Systemtheorie und soziale Bewegungen. Frankfurt am Main: Suhrkamp, 175–200.

Luhmann, Niklas, 1995a: Protestbewegungen. In: Hellmann, Kai-Uwe (Hg.), Protest: Systemtheorie und soziale Bewegungen. Frankfurt am Main: Suhrkamp, 201–215.

Luhmann, Niklas, 1995b: Die Form Person. In: Luhmann, Niklas (Hg.), Soziologische Aufklärung 6. Die Soziologie und der Mensch. Opladen: Westdeutscher Verlag, 142–154.

Luhmann, Niklas, 1995c: Kausalität im Süden. In: Soziale Systeme, Jg. 1, Heft 1, 7–28.

Luhmann, Niklas, 1996: Zeit und Gedächtnis. In: Soziale Systeme, Jg. 2, Heft 2, 307–330.

Luhmann, Niklas, 1997: Die Gesellschaft der Gesellschaft. 2 Bände, Frankfurt am Main: Suhrkamp.

Luhmann, Niklas, 1999a: Funktion und Folgen formaler Organisation. 4. Auflage. Berlin: Dunker & Humblot.

Luhmann, Niklas, 1999b: Zweckbegriff und Systemrationalität: über die Funktion von Zwecken in sozialen Systemen. 6. Auflage. Frankfurt am Main: Suhrkamp.

Luhmann, Niklas, 1999c: Gesellschaftsstruktur und Semantik: Studien zur Wissenssoziologie der modernen Gesellschaft. Band 4, 1. Auflage, Frankfurt am Main: Suhrkamp.

Luhmann, Niklas, 2000: Organisation und Entscheidung. 1. Auflage, Opladen: Westdeutscher Verlag.

Luhmann, Niklas, 2001 (1988): Erkenntnis als Konstruktion. In: Jahraus, Oliver (Hg.), Niklas Luhmann. Aufsätze und Reden. Stuttgart: Reclam Verlag, 218–242.

Luthiger, Benno, 2004: Alles aus Spaß? Zur Motivation von Open-Source-Entwicklern. In: Gehring, Robert A.; Lutterbeck, Bernd (Hg.), Open Source Jahrbuch 2004: Zwischen Softwareentwicklung und Geschäftsmodell. Berlin: Lehmanns Media.

Lutterbeck, Bernd; Bärwolff, Matthias; Gehring, Robert A. (Hg.), 2007: Open Source Jahrbuch 2007. Berlin: Lehmann Media. http://www.opensourcejahrbuch.de/download/jb2007/OpenSourceJahrbuch2007_online.pdf, 01.07.2012.

Madey, Greg (ed.): The SourceForge Research Data Archive (SRDA). University of Notre Dame. http://srda.cse.nd.edu/, 28.09.2012.

Maresch, Rudolf; Werber, Niels (Hg.), 2002: Raum – Wissen – Macht. 1. Auflage, Frankfurt am Main: Suhrkamp.

Misoch, Sabina, 2006: Online-Kommunikation. 1. Auflage, Konstanz: UVK-Verlag.

Moon, Jae Yun; Sproull, Lee, 2000: Essence of Distributed Work: The Case of the Linux Kernel. In: First Monday (Peer-Reviewed Journal of the Internet), 2000, Vol. 5, Nr. 11. http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/801/710, 18.01.2012.

Mrvar, Andrej; Batagelj, Vladimier: Pajek – Program for Large Network Analysis. Homepage: http://pajek.imfm.si, 27.09.2012.

Münker, Stefan, 2010: Was heißt eigentlich: „virtuelle Realität“? Ein philosophischer Kommentar zum neuesten Versuch der Verdopplung der Welt, In: Münker, Stefan; Roesler, Alexander (Hg.), Mythos Internet. Frankfurt am Main: Suhrkamp, 108–212.

Münker, Stefan; Roesler, Alexander (Hg.), 2010: Mythos Internet. 1. Auflage, Frankfurt am Main: Suhrkamp.

Password Safe Projektplattform: Startseite. http://sourceforge.net/projects/passwordsafe/, 26.09.2012.

Rammert, Werner (Hg.), 1995: Soziologie und künstliche Intelligenz. Produkte und Probleme einer Hochtechnologie. 1. Auflage, Frankfurt am Main: Campus Verlag.

Rammert, Werner, 1999: Virtuelle Realitäten als medial erzeugte Sonderwirklichkeiten – Veränderungen der Kommunikation im Netz der Computer, In: Faßler, Manfred (Hg.), Alle möglichen Welten. Virtuelle Realität – Wahrnehmung – Ethik der Kommunikation. München: Wilhelm Fink Verlag, 33–48.

Rammert, Werner, 2006: Technik in Aktion: Verteiltes Handeln in soziotechnischen Konstellationen. In: Rammert, Werner; Schubert, Cornelius (Hg.), Technografie. Zur Mikrosoziologie der Technik. Frankfurt am Main: Campus Verlag, 163–195.

Rammert, Werner; Schubert, Cornelius (Hg.), 2006: Technografie. Zur Mikrosoziologie der Technik. 1. Auflage, Frankfurt am Main: Campus Verlag.

Raymond, Eric S., 2001: The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. 2. (rev.) ed., Sebastopol/CA: O'Reilly.

Sandbothe, Mike, 2010: Interaktivität – Hypertextualität – Transversalität. Eine medienphilosophische Analyse des Internets. In: Münker, Stefan; Roesler, Alexander (Hg.), Mythos Internet. Frankfurt am Main: Suhrkamp, 56–82.

Schmidt, Johannes F. K., 2007: Beziehung als systemtheoretischer Begriff. In: Soziale Systeme, 2007, Jg. 13, Heft 1/2, 516–527.

Schmitt, Marco, 2009: Trennen und Verbinden. Soziologische Untersuchungen zur Theorie des Gedächtnisses. 1. Auflage, Wiesbaden: VS Verlag für Sozialwissenschaften.

Sebald, Gerd, 2008: Offene Wissensökonomie. Analysen zur Wissenssoziologie der Free/Open Source-Softwareentwicklung. 1. Auflage, Wiesbaden: VS Verlag für Sozialwissenschaften.

SourceForge.net Documentation 1a: Portal-Startseite. http://sourceforge.net, 05.01.2012.

SourceForge.net Documentation 1b: What is SOurceForge.net? http://sourceforge.net/apps/trac/sourceforge/wiki/What%20is%20SourceForge.net?, 05.01.2012

SourceForge.net Documentation 2: Tracker. http://sourceforge.net/apps/trac/sourceforge/wiki/Tracker, 05.02.2012.

SourceForge.net Documentation 3: Project Statistcs. http://sourceforge.net/apps/trac/sourceforge/wiki/Project%20statistics, 05.02.2012.

SourceForge.net Documentation 4: Developer Permissions. http://sourceforge.net/apps/trac/sourceforge/wiki/Manage%20project%20developer%20permissions, 05.02.2012.

SourceForge.net Documentation 5: Register a User Account. http://sourceforge.net/apps/trac/sourceforge/wiki/Register%20a%20user%20account, 05.02.2012.

SourceForge Research Data Archiv: Research Wiki. http://srda.cse.nd.edu/mediawiki/index.php?title=Main_Page, 26.09.2012.

Stallman, Richard, 2012a: The GNU-Manifesto. Version: 10.06.2012, http://www.gnu.org/gnu/manifesto.de.html, 09.06.2012.

Stallman, Richard, 2012b: What is Free Software? Version: 07.01.2012. http://www.gnu.org/philosophy/free-sw.en.html#History, 09.06.2012.

Stegbauer, Christian; Häußling, Roger (Hg.), 2010: Handbuch Netzwerkforschung. 1. Auflage, Wiesbaden: VS Verlag für Sozialwissenschaften.

Stewart, Katherine J.; Gosain, Sanjay, 2006: The Impact of Ideology on Effectiveness in Open Source Software Development Teams. MIS Quarterly, 2006, Vol. 30, No. 2, 291–314.

Straub, Jürgen; Weidemann, Arne; Weidemann, Doris (Hg.), 2007: Handbuch interkultureller Kommunikation und Kompetenz. Grundbegriffe – Theorien – Anwendungsfelder. 1. Auflage, Stuttgart: J. B. Metzler/Carl Ernst Poeschel Verlag.

Strübing, Jörg, 2006: Webnografie? Zu den methodischen Voraussetzungen einer ethnografischen Erforschung des Internets. In: Rammert, Werner; Schubert, Cornelius (Hg.), Technografie. Zur Mikrosoziologie der Technik. Frankfurt am Main: Campus Verlag, 249–274.

Sutter, Tilmann, 2008: „Interaktivität“ neuer Medien – Illusion und Wirklichkeit aus der Sicht einer soziologischen Kommunikationsanalyse. In: Willems, Herbert (Hg.), Weltweite Welten. Internet-Figurationen aus wissenssoziologischer Perspektive. Wiesbaden: VS Verlag für Sozialwissenschaften, 75–102.

Tacke, Veronika, 2000: Netzwerk und Adresse. In: Soziale Systeme, 2000, Jg. 6, Heft 2, 291–320.

Tacke, Veronika, 2011: Systeme und Netzwerke – oder: Was man an sozialen Netzwerken zu sehen bekommt, wenn man sie systemtheoretisch beschreibt. In: Netzwerke, Systemtheorie und Soziale Arbeit. Systemische Soziale Arbeit – Journal der dgssa, 2001, Jg. 2, Heft 2/3, 6–24.

Tannenbaum, Andy; Torvald, Linus B., 1992: Posting: Andy Tannenbaum, Feb. 6 1992, 7:17; Linus Benedict Torvalds, Feb. 7 1992, 1:55. http://groups.google.com/group/comp.os.minix/browse_thread/thread/c37da71ec2777791/7ca118a12765a380, 05.04.2012.

Trappmann, Mark; Hummell, Hans J.; Sodeur, Wolfgang (Hg.), 2011: Strukturanalyse sozialer Netzwerke. Konzepte, Modelle, Methoden. 2. Auflage. Wiesbaden: VS Verlag für Sozialwissenschaften.

Von Krogh, G.; von Hippel, E., 2003: Special issue on open source software development. Research Policy, 2003, Vol. 32, 1149–1157.

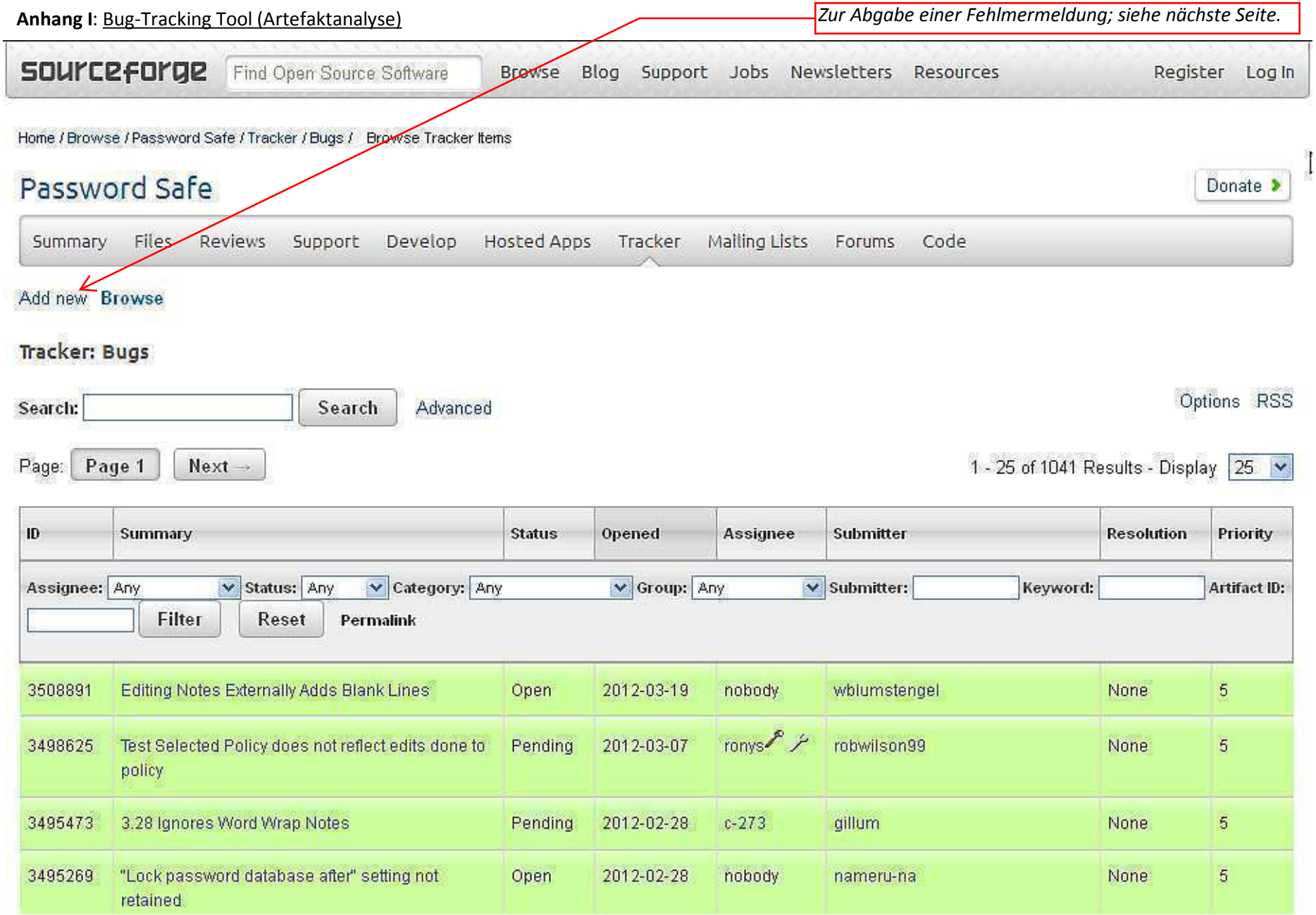
Willems, Herbert (Hg.), 2008: Weltweite Welten. Internet-Figurationen aus wissenssoziologischer Perspektive. 1. Auflage, Wiesbaden: VS Verlag für Sozialwissenschaften.

Williams, Sam, 2002: Free as in Freedom: Richard Stallman’s Crusade for Free Software. 1. Auflage, Sebastopol/CA: O’Reilly.

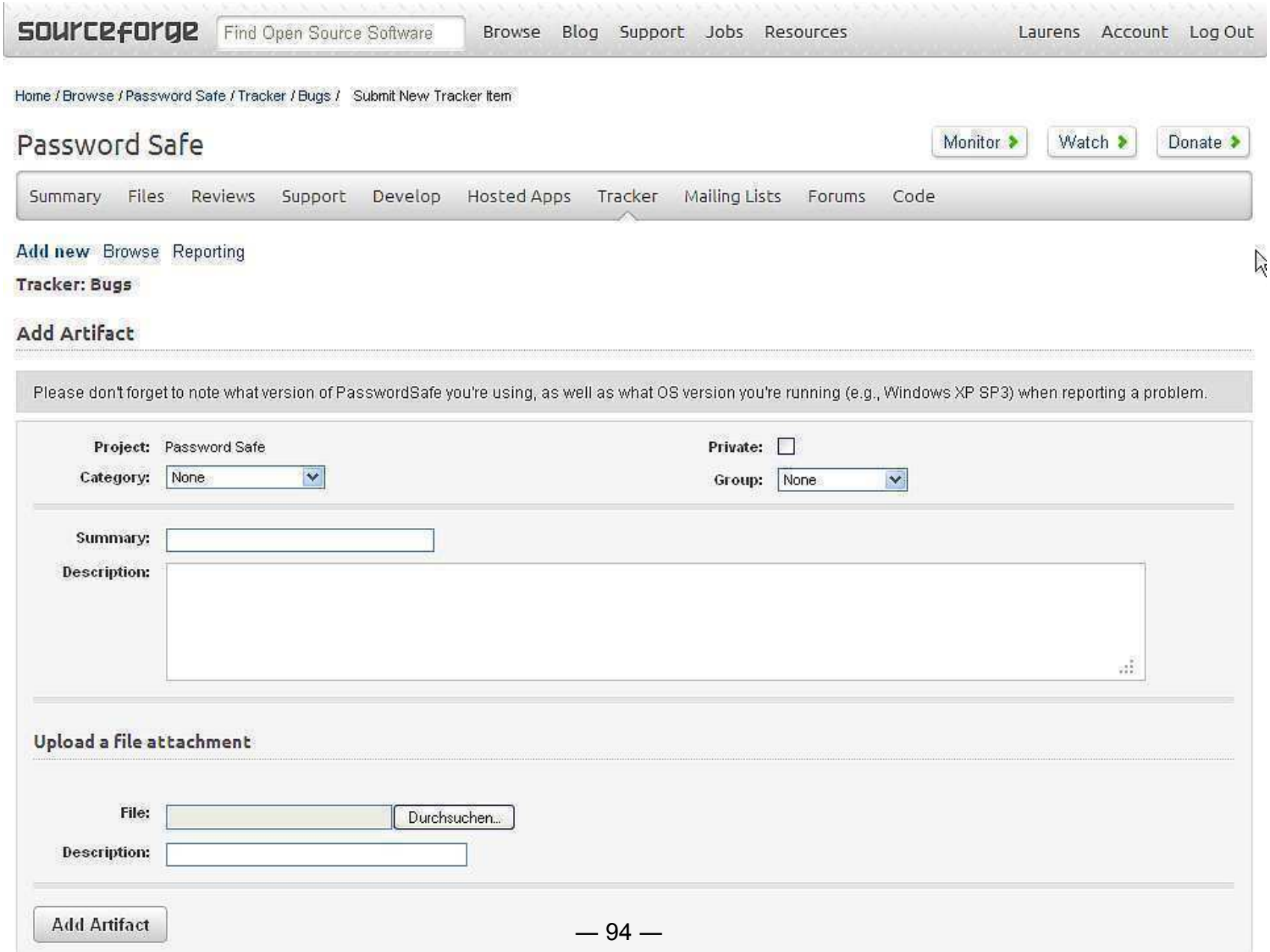
Zimmermann, T., 2004: Open Source und Freie Software – eine soziale Bewegung im virtuellen Raum? In: Gehring, Robert A.; Lutterbeck, Bernd (Hg.), Open-Source Jahrbuch 2004. Berlin: Lehmanns Media.

Anhang I: Bug-Tracking Tool (Artefaktanalyse)

Zur Abgabe einer Fehlmermeldung; siehe nächste Seite.



Eingabemaske für Fehlermeldung



Anhang II: Projektauswahl

1. Syntax zur Auswahl der Projekte unter Bedingungen der Nutzung essentieller Tools der Plattform (N=11651):

```
SELECT a. group_id, a. group_name, c. description, a. register_time

FROM   sf0511.groups a JOIN sf0511.trove_group_link b ON a. group_id = b. group_id
       JOIN sf0511.trove_cat c ON b. trove_cat_id = c. trove_cat_id

WHERE      a. use_mail != 0 AND a. use_forum !=0 AND a. use_pm != 0 AND a. use_news !=0
           AND a. use_svn != 0  AND c. description = '5 - Production/Stable' OR a. use_mail !
           = 0 AND a. use_forum !=0 AND a. use_pm != 0 AND a. use_news !=0 AND a.
           use_svn != 0  AND c. description = '6 - Mature'
```

2. Syntax zur Abfrage der Gesamtanzahl von (a) Artifacts, (b) Messages und (c) frs-Files:

(a)

```
SELECT a. group_id, d. group_artifact_id, e. count

FROM   sf0511.groups a JOIN sf0511.trove_group_link b ON a. group_id = b. group_id
       JOIN sf0511.trove_cat c ON b. trove_cat_id = c. trove_cat_id JOIN
       sf0511.artifact_group_list d ON a. group_id = d. group_id JOIN
       sf0511.artifact_counts_agg e ON e. group_artifact_id = d. group_artifact_id WHERE
       a. use_mail != 0 AND a. use_forum !=0 AND a. use_pm != 0 AND a. use_news !=0
       AND a. use_svn != 0  AND c. description = '5 - Production/Stable' OR a. use_mail !
       = 0 AND a. use_forum !=0 AND a. use_pm != 0 AND a. use_news !=0 AND a.
       use_svn != 0  AND c. description = '6 – Mature'
```

(b)

```
SELECT a. group_id, f. group_forum_id, g. count

FROM   sf0511.groups a JOIN sf0511.trove_group_link b ON a. group_id = b. group_id
       JOIN sf0511.trove_cat c ON b. trove_cat_id = c. trove_cat_id JOIN
       sf0511.forum_group_list f ON a. Group_id = f. group_id JOIN
       sf0511.forum_agg_msg_count g ON f. group_forum_id = g. group_forum_id

WHERE      a. use_mail != 0 AND a. use_forum !=0 AND a. use_pm != 0 AND a. use_news !=0
           AND a. use_svn != 0  AND c. description = '5 - Production/Stable' OR a. use_mail !
           = 0 AND a. use_forum !=0 AND a. use_pm != 0 AND a. use_news !=0 AND a.
           use_svn != 0  AND c. description = '6 – Mature'
```

(c)

```
SELECT a. group_id, COUNT (d. file_id)

FROM   sf0511.groups a JOIN sf0511.trove_group_link b ON a. group_id = b. group_id
       JOIN sf0511.trove_cat c ON b. trove_cat_id = c. trove_cat_id JOIN sf0511.frs_file d
       ON a. group_id = d. group_id

WHERE      a. use_mail != 0 AND a. use_forum !=0 AND a. use_pm != 0 AND a. use_news !=0
           AND a. use_svn != 0  AND c. description = '5 - Production/Stable' OR a. use_mail !
           = 0 AND a. use_forum !=0 AND a. use_pm != 0 AND a. use_news !=0 AND a.
           use_svn != 0  AND c. description = '6 - Mature' GROUP BY a. group_id
```

Unter Bedingung, dass jede Variable ≠ 0, ist N=2297.

3. Syntax zur Abfrage der Aktivität – produzierte (a) Artifacts (Eröffnet oder Geschlossen) und (b) Messages – über den Erhebungszeitraum (15.02.2011 – 15.05.2011):

(a)

```
SELECT b. group_id, COUNT (a. msg_id)

FROM   sf0511.forum a JOIN sf0511.forum_group_list b ON a. group_forum_id = b.
       group_forum_id

WHERE      a. date BETWEEN 1297767218 AND 1305453218 GROUP BY b. group_id
```

(b)

```
SELECT b. group_id, COUNT (a. artifact_id)

FROM   sf0511.artifact a JOIN sf0511.artifact_group_list b ON a. group_artifact_id = b.
       group_artifact_id

WHERE      a. open_date BETWEEN 1297767218 AND 1305453218 OR a. close_date
           BETWEEN 1297767218 AND 1305453218 GROUP BY b. group_id
```

Die Daten für die noch zur Auswahl-stehenden Projekte (N=45) wurden mittels Excel extrahiert und zugeordnet. Die SCM-Aktivitäten wurden über die Projektstatistik auf der jeweiligen Projektseite erhoben. Über die Suchfunktion der Plattform wurde überprüft, ob es „Schwesterprojekte“ (Projektverband) gibt.

Ausschlussbedingungen: Merkmal „Artifacts“, „Messages“ und „SCM“ ≠ 0. Ergebnis: N = 13.

Anhang III: Netzwerkerhebung

Syntax zur Abfrage der (a) Messages in den Foren, (b) Messages in den Fallbearbeitungssystemen (Tracker) und diesem hinzugefügten (c) Files für das Projekt Password Safe im Erhebungszeitraum (15.02.2011 – 15.05.2011):

(a)

```
SELECT b. group_forum_id, b. forum_name, b. description, a. thread_id, c. thread_topic, a. posted_by, a. msg_id

FROM sf0511.forum a JOIN sf0511.forum_group_list b ON a. group_forum_id = b. group_forum_id JOIN sf0511.forum_threadinfo c ON a. thread_id = c. thread_id

WHERE b. group_id = 41019 AND date BETWEEN 1297767218 AND 1305453218 ORDER BY b. group_forum_id
```

(b)

```
SELECT a. group_id, b. artifact_id, b. group_artifact_id, a. name, c. submitted_by, c. adddate

FROM sf0511.artifact_group_list a JOIN sf0511.artifact b ON a. group_artifact_id = b. group_artifact_id JOIN sf0511.artifact_message c ON b. artifact_id = c. artifact_id

WHERE a. group_id = 41019 AND c. adddate BETWEEN 1297767218 AND 1305453218
```

(c)

```
SELECT a. group_id, b. artifact_id, b. group_artifact_id, a. name, d. submitted_by

FROM sf0511.artifact_group_list a JOIN sf0511.artifact b ON a. group_artifact_id = b. group_artifact_id JOIN sf0511.artifact_file d ON b. artifact_id = d. artifact_id

WHERE a. group_id = 41019 AND d. adddate BETWEEN 1297767218 AND 1305453218
```

Die Beiträge der Entwickler zum Quellcode lassen sich nicht über die Datenbank der Plattform abfragen. Da jedoch um eine „offene“ Softwareproduktion handelt, ist der Quellcode und sein Entwicklungsverlauf im Source-Code-Management-Tool der Projekte mittels des jeweiligen Clients einsehbar. Auf diese Weise lassen sich Änderungen, Beiträge und Löschungen (zeitlich) im Protokol (Log) nachvollziehen und dem Absender zuordnen (Küng et al. 2010). Jede erfasste Transaktion wurde im vorliegenden Fall als Kommunikationsakt erfasst und der Datengrundlage des Netzwerkes hinzugefügt. Einträge in den Mailing-Listen und auf dem News-Board wurden direkt auf der Projektseite abgefragt.

Angang IV: Kennwerte 2-Mode Network (bereinigt)

Akteure:	47
Tools:	7
Events:	59 (269)*
Mitteilungen	223 (434)*

Gesamtmittelwerte:

Durchschnittliche Anzahl an Mitteilungen je Akteur:	4,74	(9,23)*
Durchschnittliche Anzahl an Events je Akteur:	1,26	(5,72)*

Häufigkeitsverteilungen Mitteilungen:**

Anzahl Mitteilungen	Freq	Freq%	CumFreq	CumFreq%
1	17	36,17	17	36,17
2	15	31,91	32	68,09
3	3	6,38	35	74,47
4	1	2,13	36	76,60
5	1	2,13	37	78,72
6	2	4,26	39	82,98
7	1	2,13	40	85,11
11	2	4,26	42	89,36
16	1	2,13	43	91,49
17	1	2,13	44	93,62
31	1	2,13	45	95,74
107	1	2,13	46	97,87
157	1	2,13	47	100,00
Sum	47	100,00		

Häufigkeitsverteilung Tools:

Anzahl Tools	Freq	Freq%	CumFreq	CumFreq%
1	40	85,11	40	85,11
2	3	6,38	43	91,49
3	2	4,26	45	95,74
6	2	4,26	47	100,00
Sum	47	100,00		

* Das CVS-Tool wird aufgrund seines besonderen Charakters in den Beitragsmöglichkeiten und der fehlenden Differenzierungsmöglichkeiten zwischen Event und Mitteilung in die Berechnung der Durchschnittswerte in spezieller Form einbezogen: Auf Ebene der Events wird es sowohl als Einmalwertung einbezogen (sowie alle Beiträge als Events gewertet) und auf Ebene der Mitteilungen werden die Beiträge zum CVS sowohl ausgeschlossen (als auch in Summe addiert).
** Inklsuvie der Mitteilungen im CVS.

Häufigkeitsverteilung Events:***

Anzahl Events		Freq	Freq%	CumFreq	CumFreq%
	1	39	82,98	39	82,98
	2	3	6,38	42	89,36
	5	1	2,13	43	91,49
	8	1	2,13	44	93,62
	16	1	2,13	45	95,74
	28	1	2,13	46	97,87
	43	1	2,13	47	100,00
Sum		47	100,00		

Durchschnittswerte innerhalb der Tools:****

Tool	**Beteiligte Akteure**	**Mitteilungen**	**Events**	-	**Mitteilungen/ Akteur**	**Mitteilungen/ Event**
Bug	11,00	35,00	9,00		3,18	3,89
Discussion	11,00	34,00	10,00		3,09	3,40
FRequest	8,00	71,00	18,00		8,88	3,94
Help	19,00	56,00	14,00		2,95	4,00
SRequest	7,00	25,00	6,00		3,57	4,17
Linux	2,00	2,00	1,00		1,00	2,00
Ø	**9,67**	**37,17**	**9,67**		**-**	**3,84**
inklusive:						
CVS	6,00	211,00	211,00		35,17	1,00 (211,00)
Ø	**9,14**	**62,00**	**38,43**		**-**	**1,61 (7,36)**

*** Das CVS wird als 1 Event behandelt (s.o.).

**** In Klammern die entsprechenden Ergebnisse bei 1-Event-Wertung des CVS-Tools. Summierung der durchschnittlichen Mitteilungen je Akteur hier nicht möglich, da Mehrfachzählung der beteiligten Akteure in den verschiedenen Tools gegeben; die Kennzahl findet sich oben.

Degree und Zentralität

Degree-Zentralisierung:	0.77
Closeness-Zentralisierung:	0.74
Betweenness-Zentralisierung:	0.73

	In-Degree	Degree	Closeness	Betweeness
User1	1	0.0217390	0.3565890	0.0000000
Developer1	5	0.1086957	0.5287356	0.0000000
Admin	24	0.5217391	0.6764706	0.3462158
User4	2	0.0434783	0.4646465	0.0000000
User5	2	0.0434783	0.5000000	0.0000000
User6	3	0.0652174	0.5054945	0.0000000
User7	1	0.0217391	0.4600000	0.0000000
User9	6	0.1304348	0.5348837	0.0085346
User10	1	0.0217391	0.4600000	0.0000000
User11	2	0.0434783	0.4646465	0.0000000
User12	1	0.0217391	0.4600000	0.0000000
User13	2	0.0434783	0.4646465	0.0000000
Developer2	37	0.8043478	0.8363636	0.7389694
User15	2	0.0434783	0.5000000	0.0000000
Developer3	5	0.1086957	0.5287356	0.0000000
User18	3	0.0652174	0.5054945	0.0000000
User19	2	0.0434783	0.5000000	0.0000000
User20	1	0.0217391	0.4070796	0.0000000
Developer4	5	0.1086957	0.5287356	0.0000000
User21	1	0.0217391	0.4600000	0.0000000
User22	1	0.0217391	0.3565891	0.0000000
User23	1	0.0217391	0.4070796	0.0000000
User24	1	0.0217391	0.4600000	0.0000000
Developer5	8	0.1739130	0.5476190	0.0869565
User26	2	0.0434783	0.4646465	0.0000000
User28	1	0.0217391	0.4600000	0.0000000
User29	3	0.0652174	0.5054945	0.0000000
User30	3	0.0652174	0.5054945	0.0000000
User31	2	0.0434783	0.5000000	0.0000000
User32	2	0.0434783	0.5000000	0.0000000
User33	1	0.0217391	0.4070796	0.0000000
User35	1	0.0217391	0.4070796	0.0000000
User36	1	0.0217391	0.4600000	0.0000000
User37	1	0.0217391	0.4600000	0.0000000
User38	1	0.0217391	0.4600000	0.0000000
User40	1	0.0217391	0.4070796	0.0000000
User44	2	0.0434783	0.4646465	0.0000000
User45	1	0.0217391	0.4600000	0.0000000
User46	1	0.0217391	0.4070796	0.0000000
User47	2	0.0434783	0.5000000	0.0000000
User49	1	0.0217391	0.4600000	0.0000000
User50	1	0.0217391	0.4600000	0.0000000
User52	1	0.0217391	0.4600000	0.0000000
User53	1	0.0217391	0.4070796	0.0000000
User55	1	0.0217391	0.4600000	0.0000000
User56	2	0.0434783	0.5000000	0.0000000
User58	1	0.0217391	0.4600000	0.0000000

Anhang V: Blocks & Cutpoints

Bi-Connected Components (Blocks)

Blocks:

Block 1	User4, Developer2, User44
Block 2	User7, Developer2
Block 3	User10, Developer2
Block 4	User12, Developer2
Block 5	User13, Developer2, User26
Block 6	Developer2, User21
Block 7	Developer2, User24
Block 8	Developer2, User28
Block 9	Developer2, User36
Block 10	Developer2, User37
Block 11	Developer2, User38
Block 12	Developer2, User45
Block 13	Developer2, User49
Block 14	Developer2, User50
Block 15	Developer2, User52
Block 16	Developer2, User55
Block 17	Developer2, User58
Block 18	Admin, User20
Block 19	Admin, User23
Block 20	Admin, User33
Block 21	Admin, User35
Block 22	Admin, User40
Block 23	Admin, User46
Block 24	Admin, User53
Block 25	Developer1, Admin, User5, User6, User9, User11, Developer2, User15, Developer3, User18, User19, Developer4, Developer5,User29, User30, User31, User32, User47, User56
Block 26	User22, Developer5,
Block 27	User1, Developer5,

Articulationpoints:

CutPoint	
1	Admin
2	Developer2
3	Developer5

Anhang VI: <u>Cliquen</u>

1) CLIQUES

```
------------------------------------------------------------------------------------------------

Minimum Set Size:              3
Input dataset:                 1Mode, Symmetrisch, Non-Valued

15 cliques found:        1:  Developer1 Admin Developer2 Developer3 Developer4 Developer5
                         2:  Admin User9 Developer2 Developer5
                         3:  Admin User5 Developer2
                         4:  Admin User6 User9 Developer2
                         5:  Admin Developer2 User15
                         6:  Admin Developer2 User18 User29
                         7:  Admin Developer2 User19
                         8:  Admin User9 Developer2 User30
                         9:  Admin Developer2 User31
                         10:  Admin Developer2 User32
                         11:  Admin Developer2 User47
                         12:  Admin Developer2 User56
                         13:  User4 Developer2 User44
                         14:  User9 User11 Developer2
                         15:  User13 Developer2 User26
```

Clique-by-Clique Actor Co-membership matrix

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	6	3	2	2	2	2	2	2	2	2	2	2	1	1	1
2	3	4	2	3	2	2	2	3	2	2	2	2	1	2	1
3	2	2	3	2	2	2	2	2	2	2	2	2	1	1	1
4	2	3	2	4	2	2	2	3	2	2	2	2	1	2	1
5	2	2	2	2	3	2	2	2	2	2	2	2	1	1	1
6	2	2	2	2	2	4	2	2	2	2	2	2	1	1	1
7	2	2	2	2	2	2	3	2	2	2	2	2	1	1	1
8	2	3	2	3	2	2	2	4	2	2	2	2	1	2	1
9	2	2	2	2	2	2	2	2	3	2	2	2	1	1	1
10	2	2	2	2	2	2	2	2	2	3	2	2	1	1	1
11	2	2	2	2	2	2	2	2	2	2	3	2	1	1	1
12	2	2	2	2	2	2	2	2	2	2	2	3	1	1	1
13	1	1	1	1	1	1	1	1	1	1	1	1	3	1	1
14	1	2	1	2	1	1	1	2	1	1	1	1	1	3	1
15	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3

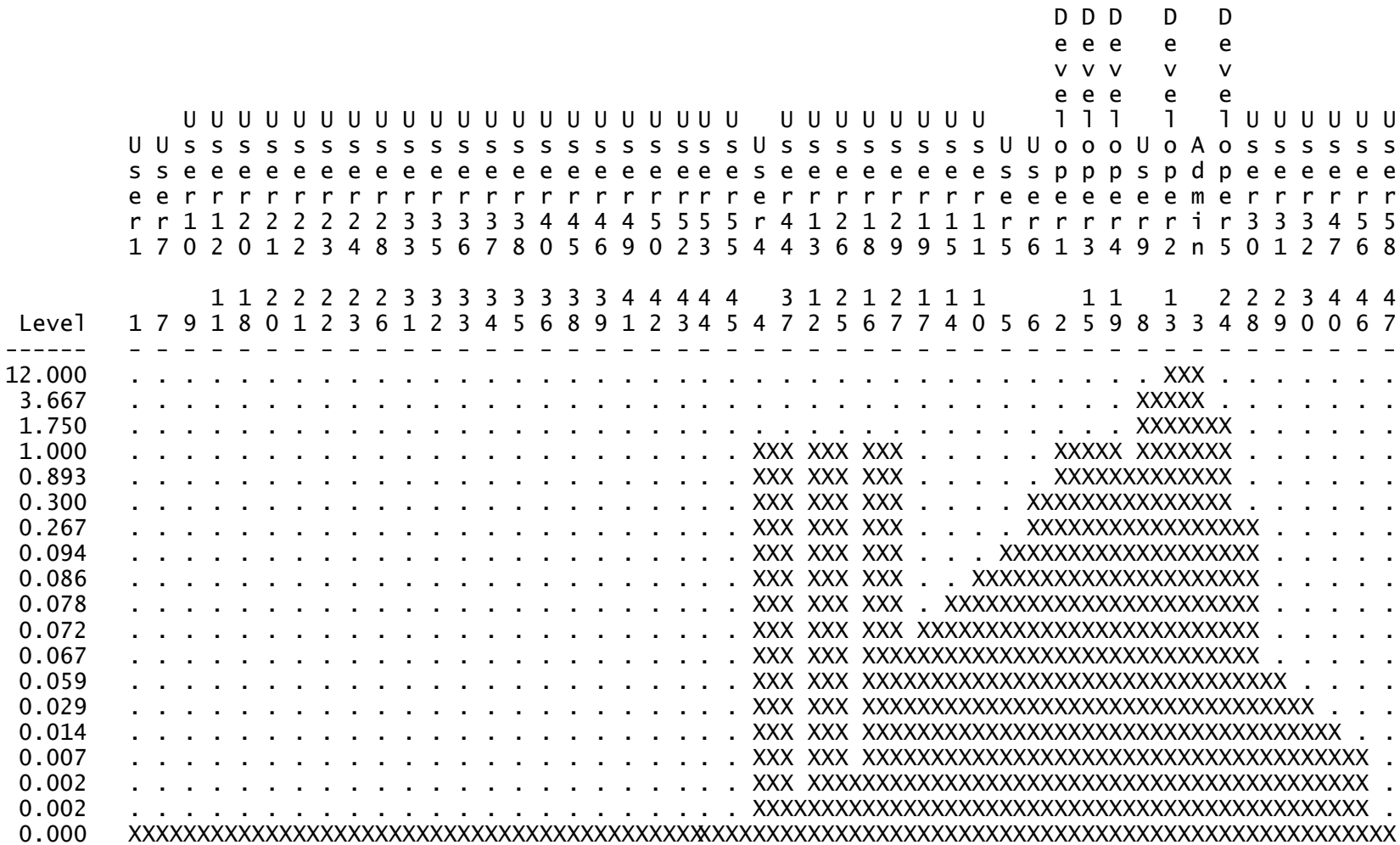
HIERARCHICAL CLUSTERING OF OVERLAP MATRIX

```
        1                   1 1 1 1 1
Level   3 3 5 6 7 2 1 8 4 9 0 1 2 4 5
-----   - - - - - - - - - - - - - - -
3.000   . . . . . XXX XXX . . . . . .
2.500   . . . . . XXXXXXX . . . . . .
2.000   . XXXXXXXXXXXXXXXXXXXXXXX . .
1.250   . XXXXXXXXXXXXXXXXXXXXXXXXX .
1.000   XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

Actor-by-Actor Clique Co-Membership Matrix

		1 Us	2 De	3 Ad	4 Us	5 Us	6 Us	7 Us	8 Us	9 Us	10 Us	11 Us	12 Us	13 De	14 Us	15 De	16 Us	17 Us	18 Us	19 De	20 Us	21 Us	22 Us	23 Us	24 De	25 Us	26 Us	27 Us	28 Us	29 Us	30 Us	31 Us	32 Us	33 Us	34 Us	35 Us	36 Us	37 Us	38 Us	39 Us	40 Us	41 Us	42 Us	43 Us	44 Us	45 Us	46 Us	47 Us
1	User1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	Developer1	0	1	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	Admin	0	1	12	0	1	1	0	3	0	0	0	0	12	1	1	1	1	0	1	0	0	0	0	2	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0
4	User4	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
5	User5	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	User6	0	0	1	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	User7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	User9	0	0	3	0	0	1	0	4	0	1	0	0	4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	User10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	User11	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	User12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	User13	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	Developer2	0	1	12	1	1	1	0	4	0	1	0	1	15	1	1	1	1	0	1	0	0	0	0	2	1	0	1	1	1	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0
14	User15	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	Developer3	0	1	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	User18	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	User19	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	User20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	Developer4	0	1	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	User21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	User22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	User23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	User24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	Developer5	0	1	2	0	0	0	0	1	0	0	0	0	2	0	1	0	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	User26	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	User28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	User29	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	User30	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	User31	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	User32	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	User33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	User35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	User36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	User37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	User38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	User40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	User44	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
38	User45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	User46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	User47	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
41	User49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
42	User50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
43	User52	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
44	User53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
45	User55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
46	User56	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
47	User58	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

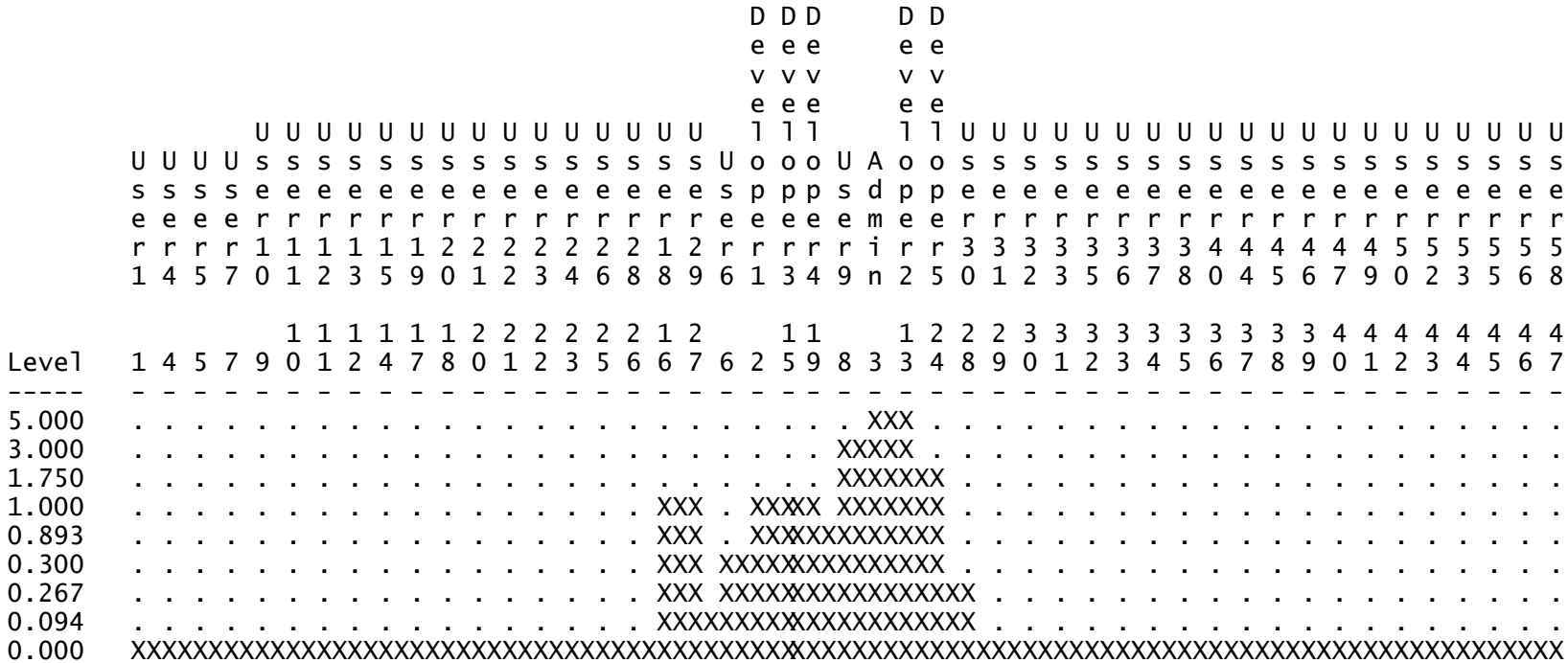
HIERARCHICAL CLUSTERING OF OVERLAP MATRIX



2) CLIQUES

Minimum Set Size: 4
Input dataset: 1Mode, Symmetrisch, Non-Valued

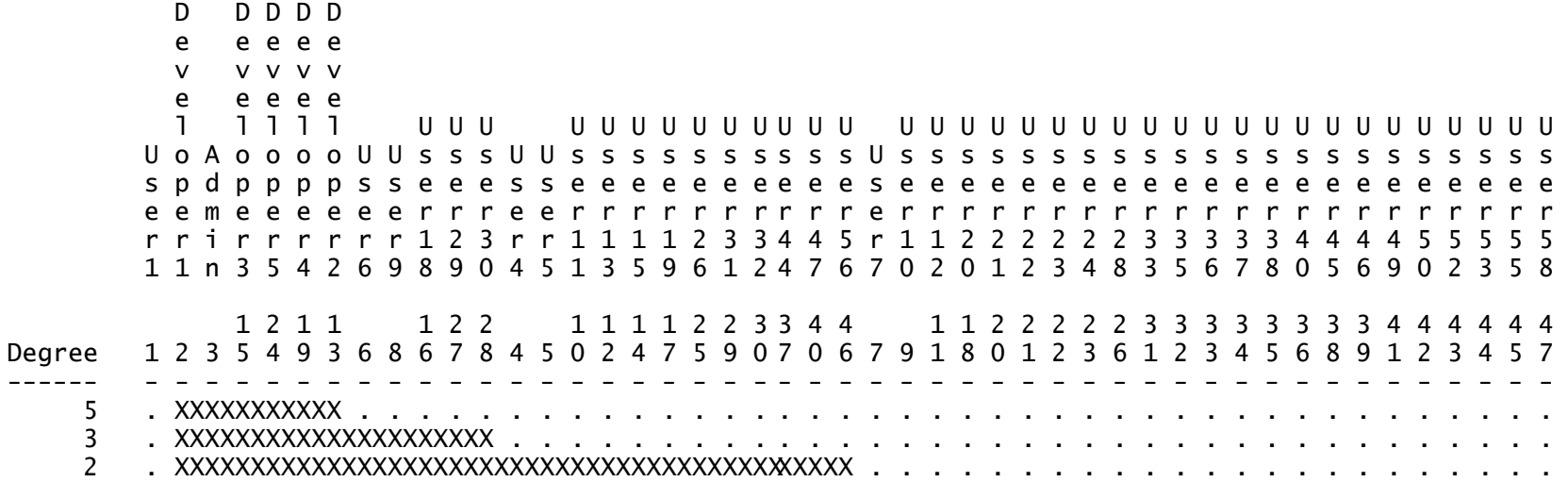
HIERARCHICAL CLUSTERING OF OVERLAP MATRIX



3) K-CORE

Input dataset: 1Mode, Symmetrisch, Non-Valued
Output partitions: KCores

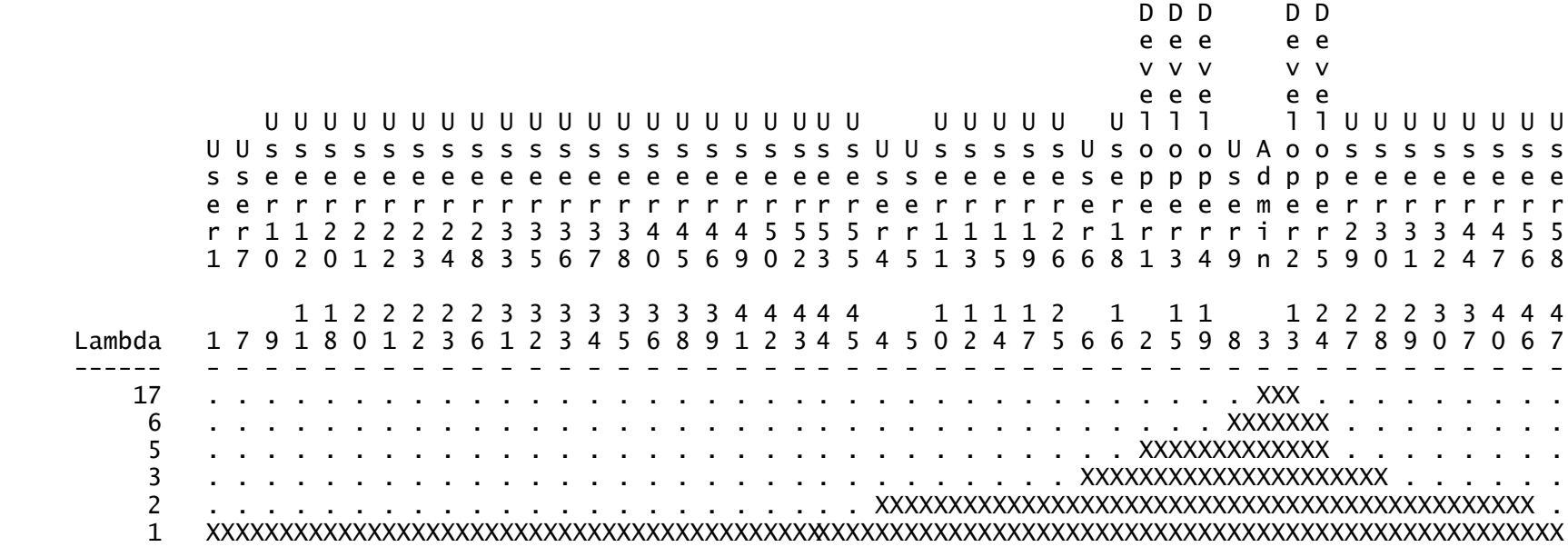
K-CORE PARTITIONING



4) LAMBDA SETS

Input dataset: 1Mode, Symmetrisch, Single

HIERARCHICAL LAMBDA SET PARTITIONS

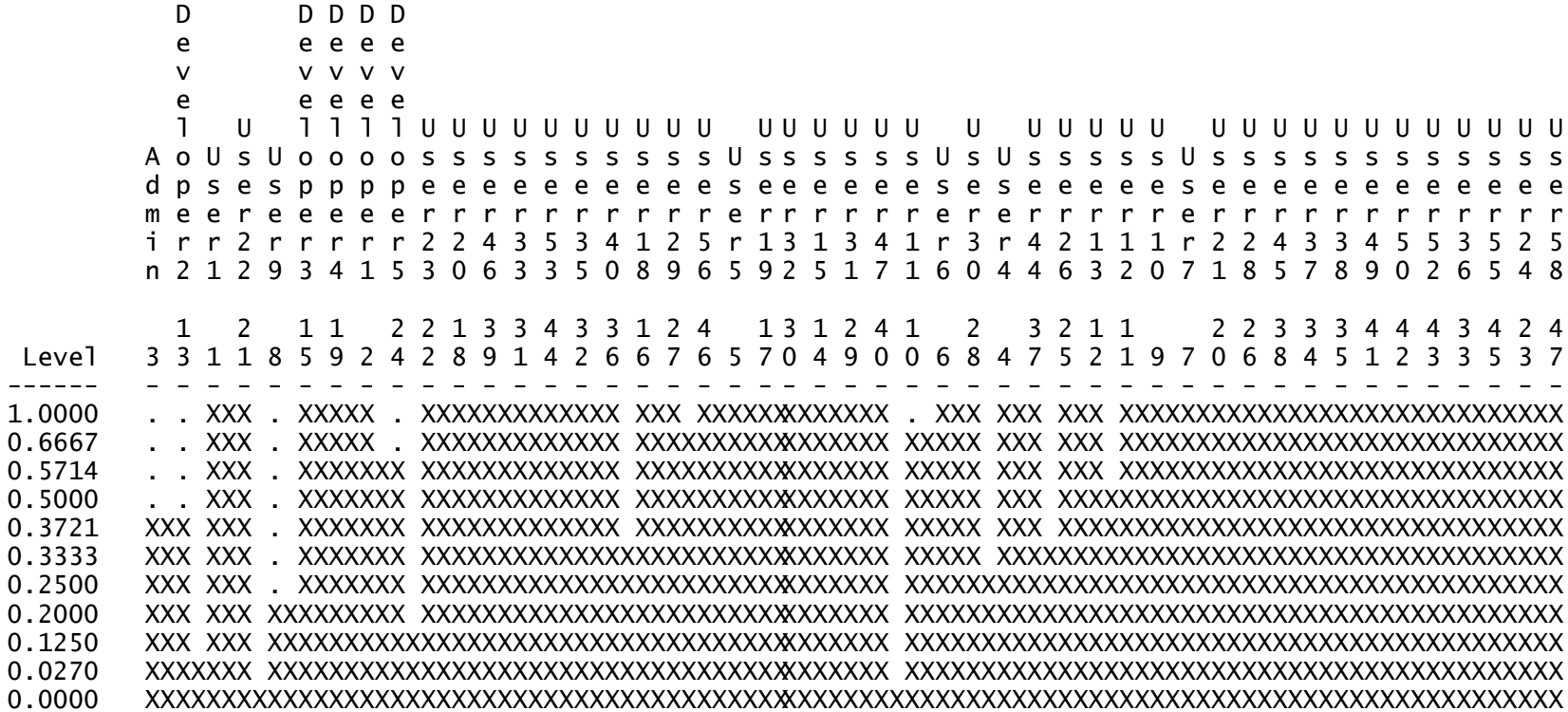


Anhang VII: Clustering

```
Netzwerk:                       1Mode,Symmetrisch,Non-Valued
Type of Data:                   Similarities; Matches
Method:                         COMPLETE_LINK (average distance)


HIERARCHICAL CLUSTERING

          D   D D D D
          e   e e e e
          v   v v v v
          e   e e e e
          l   l l l l   U   U U U U U U U U U   U  U U U U U U U   U U U U U   U U U U U U U U U U U
          o A o o o o U s U s s s s s s s s s U s U s s s s s s s U s s s s s U s s s s s s s s s s s
          p d p p p p s e s e e e e e e e e e s e s e e e e e e e s e e e e e s e e e e e e e e e e e
          e m e e e e e r e r r r r r r r r r e r e r r r r r r r e r r r r r e r r r r r r r r r r r
          r i r r r r r 1 r 3 1 2 5 1 3 3 1 4 r 2 r 3 2 5 4 2 3 4 r 4 2 1 1 1 r 2 2 4 3 3 4 5 5 3 5 2 5
          2 n 1 3 4 5 9 1 6 0 8 9 6 5 2 1 9 7 5 2 1 3 3 3 0 0 5 6 4 4 6 3 2 0 7 1 8 5 7 8 9 0 2 6 5 4 8

          1     1 1 2   1   2 1 2 4 1 3 2 1 4   2  3 2 4 3 1 3 3   3 2 1 1     2 2 3 3 3 4 4 4 3 4 2 4
 Level    3 3 2 5 9 4 8 0 6 8 6 7 6 4 0 9 7 0 5 1 1 1 2 4 6 8 2 9 4 7 5 2 1 9 7 0 6 8 4 5 1 2 3 3 5 3 7
------    - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
1.0000    . . XXXXX . . . XXX XXX XXXXXXXXXXXXX XXX XXXXXXXXXXXXX XXX XXX XXXXXXXXXXXXXXXXXXXXXXXXXX
0.9778    . . XXXXX . . XXXXX XXXXXXXXXXXXXXXXX XXX XXXXXXXXXXXXX XXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0.9556    . . XXXXX . . XXXXX XXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0.9333    . . XXXXXXX . XXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0.9111    . . XXXXXXX . XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0.8667    . . XXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0.8222    . . XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0.4667    . XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
0.2000    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX


Measures of cluster adequacy

                 1      2      3      4      5      6      7      8
             ------ ------ ------ ------ ------ ------ ------ ------
         Eta  0.227  0.251  0.272  0.338  0.612  0.637  0.948  0.828
           Q -0.014 -0.013 -0.011 -0.008 -0.002 -0.001 -0.000 -0.000
     Q-prime -0.015 -0.014 -0.012 -0.009 -0.002 -0.002 -0.000 -0.000
         E-I  0.683  0.591  0.495  0.104 -0.559 -0.632 -0.933 -0.979
```

Netzwerk: 1Mode,Symmetrisch,Non-Valued
Type of Data: Similarities; Jaccard
Method: COMPLETE_LINK (average distance)

HIERARCHICAL CLUSTERING

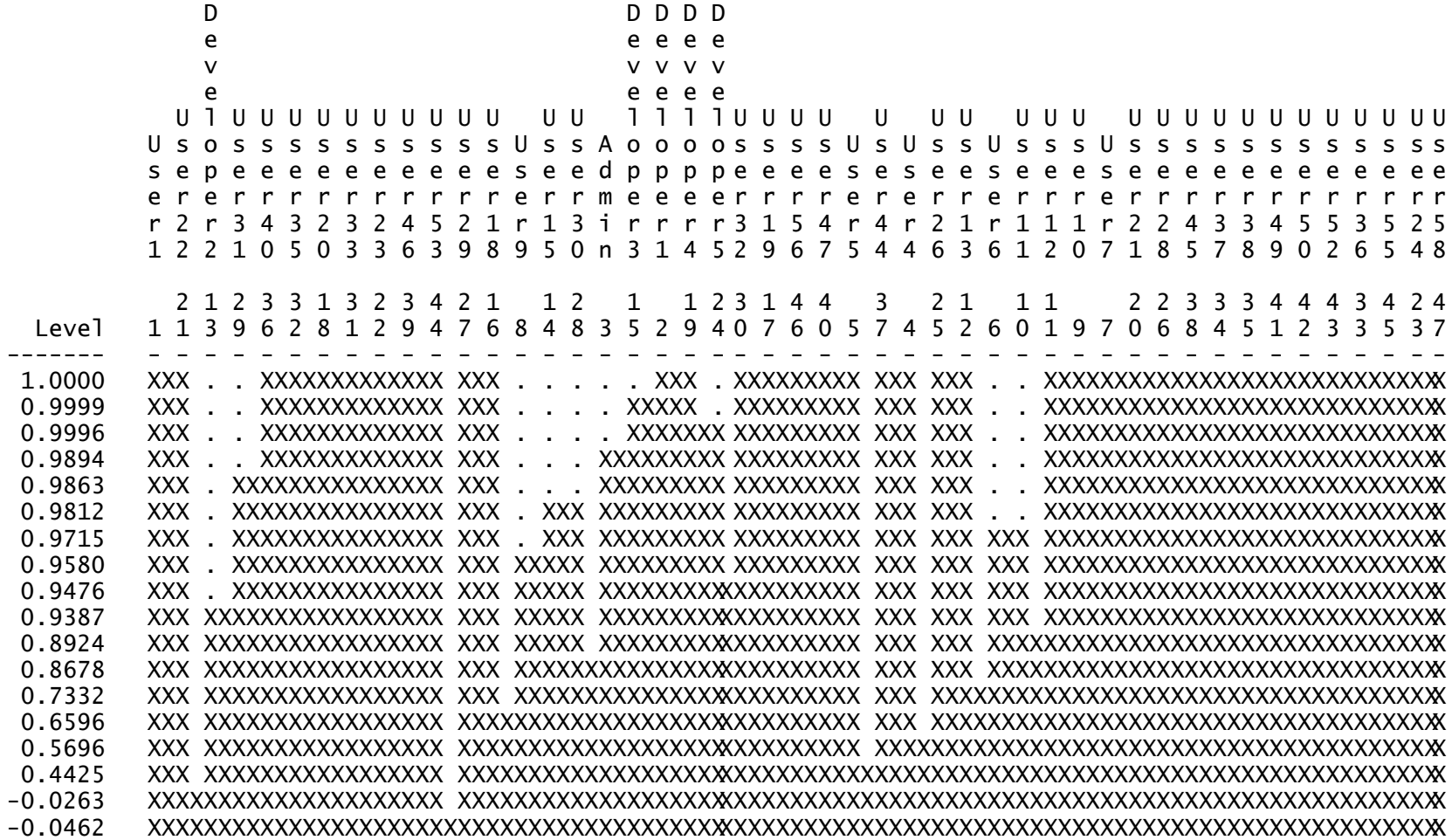


Measures of cluster adequacy

	1	2	3	4	5	6	7	8	9	10
Eta	0.804	0.810	0.810	0.790	0.789	0.755	0.721	0.718	0.648	0.640
Q	0.150	0.158	0.161	0.170	0.171	0.202	0.193	0.195	0.198	0.198
Q-prime	0.162	0.173	0.178	0.189	0.192	0.236	0.232	0.243	0.264	0.297
E-I	0.207	0.153	0.144	0.067	0.066	-0.167	-0.274	-0.280	-0.392	-0.392

Netzwerk: 1Mode,Symmetrisch,Valued
Type of Data: Similarities; Correlation
Method: COMPLETE_LINK (average distance)

HIERARCHICAL CLUSTERING



Measures of cluster adequacy

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Eta	0.435	0.439	0.444	0.451	0.462	0.464	0.465	0.468	0.504	0.514	0.548	0.579	0.601	0.601
Q	0.056	0.056	0.057	0.057	0.065	0.065	0.066	0.066	0.068	0.077	0.087	0.085	0.097	0.096
Qprime	0.059	0.060	0.060	0.061	0.070	0.071	0.072	0.073	0.076	0.087	0.099	0.099	0.116	0.120
E-I	0.546	0.540	0.530	0.518	0.495	0.492	0.489	0.483	0.406	0.381	0.293	0.207	0.118	0.060

	15	16	17
Eta	0.602	0.804	0.759
Q	0.107	0.088	0.088
Qprime	0.142	0.132	0.175
E-I	-0.023	-0.748	-0.748

Ergänzend: nur Kerngruppe

Netzwerk: 1Mode,Symmetrisch,Valued
Type of Data: Similarities; Correlation
Method: COMPLETE_LINK (average distance)

HIERARCHICAL CLUSTERING

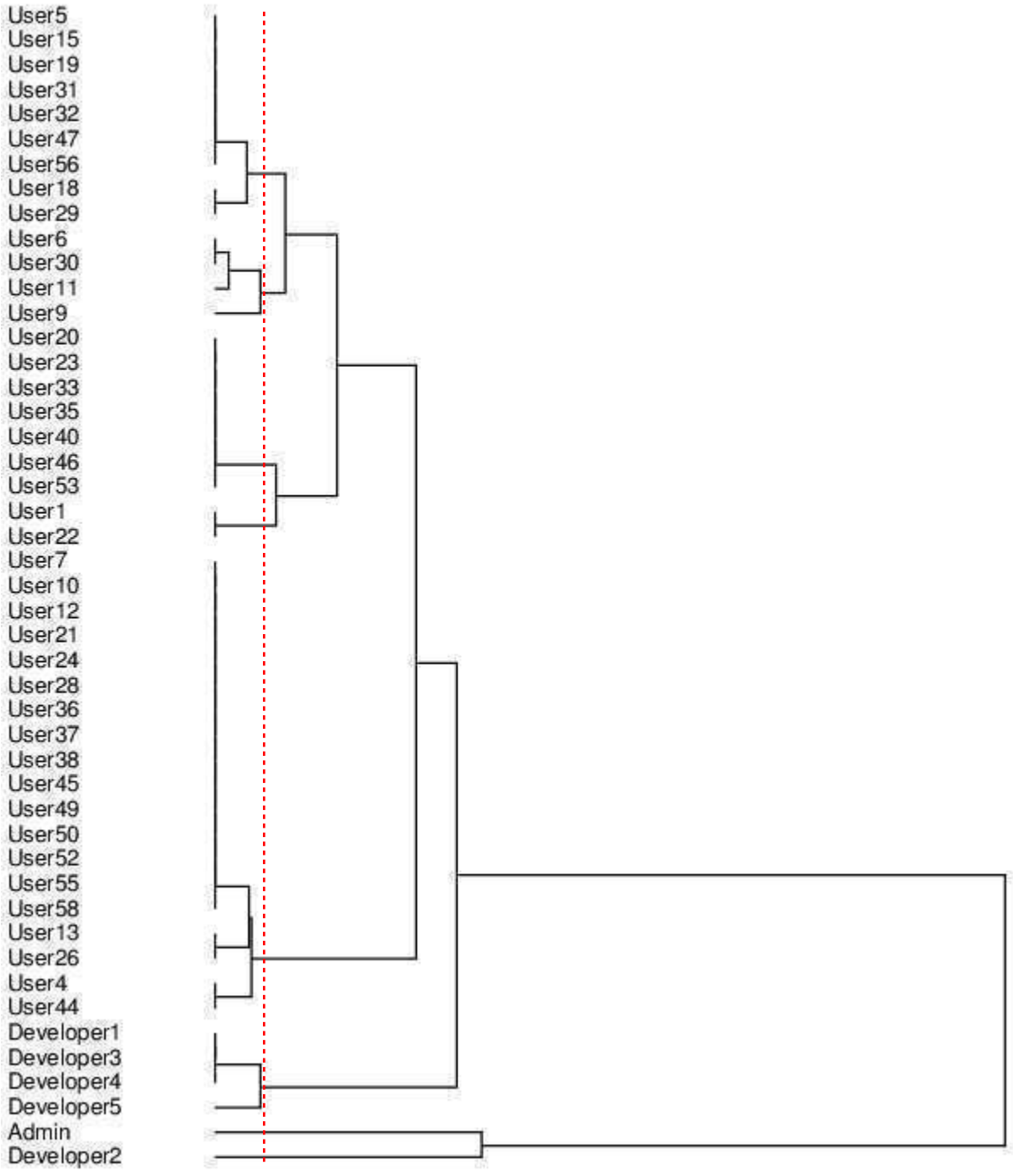
```
                                D D D D D
                                e e e e e
                                v v v v v
                                e e e e e
          U U U     U   A     l l l l l   U U U U U U
          s s s U U s s d     o o o o o U s s s s s s
          e e e s s e m e     p p p p p s e e e e e e
          r r r e e r i r     e e e e e e r r r r r r
          1 2 1 r r 3 n 2     r r r 1 5 5 1 5 9 2 7 6
          8 9 1 6 9 0   3     4 1 3 3 6 8 1 7 8 9 4 5

          1 1       1       1   1   1   1 1 1 1
  Level   0 4 6 4 5 5 2 7 9 2 1 3 3 6 8 1 7 8 9
-------   - - - - - - - - - - - - - - - - - - -
 1.0000   XXX . . . . . . . . . . . . . XXXXXXX
 0.9994   XXX . . . . . . XXX . . . . . XXXXXXX
 0.9980   XXX . . . . . . XXX . . XXX . XXXXXXX
 0.9979   XXX . . . . . . XXX XXX XXX . XXXXXXX
 0.9906   XXX . . . . . . XXXXXXX XXX . XXXXXXX
 0.9885   XXX . . . . XXX XXXXXXX XXX . XXXXXXX
 0.9846   XXX . . . . XXX XXXXXXX XXX XXXXXXXXX
 0.9737   XXX . XXX . XXX XXXXXXX XXX XXXXXXXXX
 0.9598   XXX . XXX . XXXXXXXXXXX XXX XXXXXXXXX
 0.9542   XXX . XXXXX XXXXXXXXXXX XXX XXXXXXXXX
 0.9010   XXX . XXXXX XXXXXXXXXXX XXXXXXXXXXXXX
 0.8392   XXX XXXXXXX XXXXXXXXXXX XXXXXXXXXXXXX
 0.7234   XXX XXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXX
 0.4435   XXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXX
-0.0126   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

Pajek Dissimilarity

N(v) are All neighbours of vertex v.
| stands for set cardinality.
+ stands for symmetric sum of sets.
1st maxdegree and 2nd maxdegree stand for largest degree and second largest degree in network, respectivelly.

$$d1(u,v) = | N(u) + N(v) | / (\text{1st maxdegree} + \text{2nd maxdegree})$$

$$d3(u,v) = \frac{| N(u) + N(v) |}{|N(u)| + |N(v)|}$$



Anhang VIII: Blockmodeling

Ausgangseinteilung:

Cluster	Knoten
A	Admin
B	Developer2
C	Developer1, Developer3, Developer4, Developer5
D	User6, User9, User11, User30
E	User5, User15, User18, User19, User29, User31, User32, User47, User56
F	User1, User22
G	User20, User23, User33, User35, User40, User46, User53
H	User4, User7, User10, User12, User13, User21, User24, User26, User28, User36, User37, User38, User44, User45, User49, User50, User52, User55, User58

Berechnungsgrundlage ist das Blockmodel-Verfahren in Pajek zur Optimierung bestehender Partitionen unter den Bedingungen regulärer Äquivalenz; ergänzend wurden offene Verfahren (Random) sowie benutzerdefinierte Blockeinteilungen und Beziehungsbedingungen herangezogen. Die folgenden Ausführungen beschränken sich auf die Analyseschritte, die für das abschließende Ergebnis besondere Relevanz besitzen.

1) Ausgangseinteilung

Partition: 1=A, 2=B, 3=C, 4=D, 5=E, 6=F, 7=G, 8=H

Regular

```
Initial Image Matrix:

     1   2   3   4   5   6   7   8
1    -  com com com com   -  com   -
2  com   -  com com com   -   -  com
3  com com com   -   -   -   -   -
4  com com   -  reg   -   -   -   -
5  com com   -   -   -   -   -   -
6    -   -   -   -   -   -   -   -
7  com   -   -   -   -   -   -   -
8    -  com   -   -   -   -   -   -

Initial Error Matrix:

        1    2    3    4    5    6    7    8
  1     0    0    0    1    0    0    0    0
  2     0    0    0    0    0    0    0    0
  3     0    0    0    1    0    2    0    0
  4     1    0    1    0    0    0    0    0
  5     0    0    0    0    2    0    0    0
  6     0    0    2    0    0    0    0    0
  7     0    0    0    0    0    0    0    0
  8     0    0    0    0    0    0    0    4

Initial error = 14.000
```

2) Fusion Cluster A+B, F+G+H

Partition: 1=A+B, 2=C, 3=D, 4=E, 5=F+G+H

Initial Image Matrix:

	1	2	3	4	5
1	com	com	reg	com	reg
2	com	com	–	–	–
3	reg	–	reg	–	–
4	com	–	–	–	–
5	reg	–	–	–	–

Initial Error Matrix:

	1	2	3	4	5
1	0	0	0	0	4
2	0	0	1	0	2
3	0	1	0	0	0
4	0	0	0	2	0
5	4	2	0	0	4

Initial error = 20.000

3) Fusion A+B, E+F+G+H

Partition: 1=A+B, 2=C, 3=D, 4=E+F+G+H

Initial Image Matrix:

	1	2	3	4
1	com	com	reg	reg
2	com	com	–	–
3	reg	–	reg	–
4	reg	–	–	–

Initial Error Matrix:

	1	2	3	4
1	0	0	0	4
2	0	0	1	2
3	0	1	0	0
4	4	2	0	6

Initial error = 20.000

→ Optimierung mit Umcodierung der Clusterzugehörigkeit von Developer5 zu Cluster 1:

```
Initial Image Matrix:

     1   2   3   4
1  com com reg reg
2  com com  -   -
3  reg  -  reg  -
4  reg  -   -   -


Initial Error Matrix:

        1    2    3    4
1       0    0    0    0
2       0    0    0    0
3       0    0    0    0
4       0    0    0    6

Initial error = 6.000
```

Alternativ:

4) Fusion A+B,C+D, E+F+G+H
Partition: 1=A+B, 2=C+D, 3=E+F+G+H

```
Initial Image Matrix:

     1   2   3
1  com reg reg
2  reg reg  -
3  reg  -   -


Initial Error Matrix:

        1    2    3
1       0    0    4
2       0    0    2
3       4    2    6

Initial error = 18.000
```

→ Optimierung mit Umcodierung der Clusterzugehörigkeit von Developer5 zu Cluster 1:

```
Initial Image Matrix:

     1   2   3
1  com reg reg
2  reg reg  -
3  reg  -   -

Initial Error Matrix:

        1    2    3
1       0    0    0
2       0    0    0
3       0    0    6

Initial error = 6.000
```

Ergänzungen:

- Fusion A+B, C+D+E, F+G+H
  Partition: 1=A+B, 2=C+D+E, 3=F+G+H

```
Initial Image Matrix

     1   2   3
1  com reg reg
2  reg  -   -
3  reg  -   -

Initial error = 38.0(
```

```
Initial Error Matrix:

        1    2    3
1       0    0    4
2       0   22    2
3       4    2    4
```

- Partition Developer vs. User

```
Initial Image Matrix:

    1   2
1   -   -
2   -  com

Initial error = 120.000
```

```
Initial Error Matrix:

        1    2
1      12   54
2      54    0
```

- Partition Cliquen

```
Initial Image Matrix:

    1   2   3   4   5
1   -   -   -   -   -
2   -   -   -   -   -
3   -   -  reg  -   -
4   -   -   -   -   -
5   -   -   -   -  com

Initial error = 114.000
```

```
Initial Error Matrix:

        1    2    3    4    5
1       0    0    0    0   24
2       0    4    1    0   19
3       0    1    0    0   11
4       0    0    0    0    0
5      24   19   11    0    0
```

Anhang IX: Rollenanalyse

GOULD & FERNANDEZ BROKERAGE MEASURES

Input dataset: 1Mode, directed, NonValued
Method: WEIGHTED -> Un-normalized Brokerage Scores

Hinweis: Die Ergebnistabellen werden unter Auslassung der Knoten ohne Kennwertausprägung zusammengefasst!!!

1) Partition User vs. Developer (2 classes):

	1 Coordinat	2 Gatekeepe	3 Represent	4 Consultan	5 Liaison	6 Total
User1	0	0	0	0	0	0
...						
User9	2.667	1.667	1.667	0	0	6.000
...						
User53	0	0	0	0	0	0
Admin	0	57.667	57.667	272.667	0	388.000
Developer3	0	0	0	0	0	0
Developer4	0	0	0	0	0	0
Developer1	0	0	0	0	0	0
Developer2	0	121.667	121.667	914.667	0	1158.000
Developer5	0	11.000	11.000	6.000	0	28.000

2) Partition Blockmodel-Einteilung (4 classes)

	1 Coordinat	2 Gatekeepe	3 Represent	4 Consultan	5 Liaison	6 Total
Developer5	0	4.000	4.000	2.000	18.000	28.000
Developer2	0	44.167	44.167	717.667	352.000	1158.000
Admin	0	19.167	19.167	203.667	146.000	388.000
Developer3	0	0	0	0	0	0
Developer1	0	0	0	0	0	0
Developer4	0	0	0	0	0	0
User30	0	0	0	0	0	0
User6	0	0	0	0	0	0
User9	2.667	1.667	1.667	0	0	6.000
User11	0	0	0	0	0	0
User7	0	0	0	0	0	0
...						
User58	0	0	0	0	0	0

3) Partition Blockmodel mit Fusion von Block A und Block B (3 classes):

	1 Coordinat	2 Gatekeepe	3 Represent	4 Consultan	5 Liaison	6 Total
Developer5	0	11.000	11.000	2.000	4.000	28.000
Developer1	0	0	0	0	0	0
Admin	0	57.667	57.667	203.667	69.000	388.000
Developer3	0	0	0	0	0	0
Developer2	0	121.667	121.667	717.667	197.000	1158.000
Developer4	0	0	0	0	0	0
User30	0	0	0	0	0	0
User6	0	0	0	0	0	0
User9	2.667	1.667	1.667	0	0	6.000
User11	0	0	0	0	0	0
User7	0	0	0	0	0	0
...						
User58	0	0	0	0	0	0

Lebenslauf

Zur Person

Name	Laurens Lauer
Geburtsdaten:	15. März 1985 in Essen
Staatsangehörigkeit:	deutsch
Kontakt:	LaurensLL@hotmail.de

Studium

seit 10/2009	Masterstudium Soziologie an der Hauptuniversität Wien, Österreich.
10/2005 – 08/2009	Bachelorstudium Sozialwissenschaften an der Phillips-Universität Marburg, Deutschland.
03/2008 – 08/2008	Auslandsemester an der Karls-Universität Prag, Tschechien.

Schulische Ausbildung

2001 – 2005	Burggymnasium Essen, Abschluss: Abitur.
1991 – 2001	Freie Waldorfschule Essen.

Praktika und berufliche Erfahrungen

seit 2007	Begleitender wissenschaftlicher Mitarbeiter der Unternehmensberatung Prof. Dr. Sven Grote.
02/2007 – 04/2007	Praktikum am Institut für Arbeitswissenschaften an der Universität Kassel, Deutschland.

Publikationen

Branscheidt, M., Lauer, L., 2012: Romantische Liebe in der neurobiologischen Forschung. In: Mixa, E., Vogel, P. (Hg.), E-Motions. Transformationsprozesse in der Gegenwartskultur. Wien: Verlag Turia + Kant.

Grote, S., Kauffeld, S., Denison, K., Billich-Knapp, M., Lauer, L., Frieling, E., 2012: Kompetenzen und deren Management: ein Überblick. In: Grote, S., Kauffeld, S., Frieling, E. (Hg.), Kompetenzmanagement. Grundlagen und Praxisbeispiele. 2. überarb. Auflage, Stuttgart: Schäffer-Poeschel Verlag, 15-34.

Grote, S., Hering, V., Casper, V, Lauer, L., (in Druck): Zwischen Stabilität und Dynamik: Perspektiven des Balance-Modells der Führung. In: Grote, S. (Hg.), Die Zukunft der Führung. Berlin: Springer-Verlag.

Hertel, G., Lauer, L., (in Druck): Führung auf Distanz und E-Leadership – die Zukunft der Führung? In: Grote, S. (Hg.), Die Zukunft der Führung. Berlin: Springer-Verlag.

Kauffeld, S., Grote, S. & Lauer, L. (in Druck): Gruppendiagnostik. In: Werner, C. & Elbe, M. (Hrsg.). Handbuch Organisationsdiagnose. München: Herbert Utz.

Zusammenfassung

Die kooperative Entwicklung und Produktion freier/offener Computersoftware im Internet ist ein noch recht junges, aber von der Öffentlichkeit zunehmend beachtetes Phänomen sozialer Wissensproduktion. Unter dem Label „Free Software“ und „Open Source“ schließen sich global verteilte Akteure in einer Vielzahl von Projekten zusammen und widmen sich auf unbezahlter, freiwilliger Basis der gemeinsamen Herstellung von Software. Ihre Zusammenarbeit, meist ausschließlich über schriftliche Kommunikation koordiniert, folgt dabei den Maximen des unbeschränkten Austausches von Informationen und der Einbindung jedes Interessierten.

Die vorliegende explorative Fallanalyse untersucht vor diesem Hintergrund die organisationalen Strukturen und Prozesse eines Projekts der freien/offenen Softwareentwicklung, um diese unter dem Aspekt sozialer Ordnungsleistung einem grundlegenden Verständnis zugänglich zu machen. In Rückgriff auf die bestehende Forschung wird zunächst das gesellschaftliche Feld freier/offener Softwareentwicklung als kontextualer Rahmen des kooperativen Zusammenschlusses erarbeitet und dargestellt. Darauf aufbauend folgt die Analyse der sozialen Organisationsformen eines ausgewählten Projekts anhand der exemplarischen Artefaktanalyse eines virtuellen Kommunikationstools und der Beziehungsstrukturen des Kommunikationsnetzwerkes des Projekts auf der Internetplattform sourceforge.net. Die Ergebnisse werden aus einer systemtheoretischen Perspektive schrittweise zusammengefasst und interpretiert.

Die Fallanalyse zeigt eine grundlegende soziale Abgrenzung von Entwicklern und Anwendern (Leistungs- und Publikumsrolle), die sich in einem ausgeprägten Zentrum-Peripherie-Verhältnis organisieren. Ihre Kooperation realisiert sich in Form einer stark sachlich strukturierten Systemdifferenzierung, vor deren Hintergrund sich als emergentes Ordnungsprinzip eine abstrahierende Zwecksetzung bzw. —programmierung freier/offener Softwareentwicklung rekonstruieren und zu einem funktionalen Verständnis der System- und Relationsausprägungen heranziehen lässt. Die Ergebnisse ermöglichen wesentliche Einsichten in die organisationalen Strategien gemeinsamer Wissensproduktion im Internet.

Abstract

The cooperative development and production of free/open computer software on the internet is a relatively recent, but public acclaimed phenomenon of social production of knowledge. Under the label of “free software” and “open source” global distributed actors are involved in projects and devote themselves to the production of software on an unpaid, voluntary basis. Their cooperation coordinates (mostly) exclusively through written communication and follows the maxims of open access to and unlimited exchange of information as well as the integration of each person interested.

This case analysis examines the organizational structures and processes of a free/open software project to allow a fundamentally understanding under the aspect of social ordner. With recourse to existing research, at first the social field of free/open software development will be elaborated and presented as a contextual framework of the cooperative mergers. Based on this, the social organization of a project on the internet platform sourceforge.net will be analyzed by means of virtual communication tools and the structure of relationship of the communication network. The results are gradually summarized and interpreted from a social systems theory perspective.

The case study shows a basic social distinction between developers and users (performance and audience role), organized in a pronounced centre-periphery relationship. Their cooperation is realized in form of a strongly factual structured system differentiation. Against this background an abstractive purpose-driven setting or programming of free/open software development can be reconstructed as an emergent principle of order and can be adducted for a functional understanding of system relation characteristics. These results provide important insights into the organizational strategies of joint production of knowledge on the Internet.