Forschungssoftware in der Kommunikations- und Medienwissenschaft: Stand, Herausforderungen und Perspektiven

März 2021

DGPuK-AG Forschungssoftware

Andreas Hepp, Florian Hohmann, Alessandro Belli, Karin Boczek, Mario Haim, Annett Heft, Jakob Jünger, Pascal Jürgens, Erik Koenen, Gerret von Nordheim, Lars Rinsdorf, Liane Rothenberger, Tim Schatto-Eckrodt und Julian Unkel

Inhalt:

1. Die Kommunikations- und Medienwissenschaft im „digitalen Wandel“

Die letzten zwei Jahrzehnte haben viel Veränderung für die Kommunikations- und Medienwissenschaft gebracht: Das Internet und die Digitalisierung haben den Gegenstandsbereich der Kommunikations- und Medienwissenschaft grundlegend verändert. Weltweit haben sich digitale Plattformen und soziale Medien durchgesetzt und auch die „traditionellen“ Medien der öffentlichen Kommunikation wurden zu digitalen. Dieser „digitale Wandel“ bedeutet das Entstehen eines „hybriden Mediensystems“ (Chadwick, 2017), in dem „alte“ und „neue“ Medien in engen Wechselbeziehungen stehen und sich die Gesamtdynamiken gesellschaftlicher Kommunikation ändern. Bestehende Vorstellungen von Massenkommunikation und Individualkommunikation gerieten so unter Druck, wie sich auch der generelle **Begriffs- und Theorieapparat** der Kommunikations- und Medienwissenschaft veränderte. Grundlegend dabei ist, dass Medien durch ihre Digitalisierung nicht mehr nur Mittel der Kommunikation sind, sondern sie dienen in jedem Akt der Kommunikation immer auch der Datengenerierung. Dies eröffnet nicht nur neue Geschäftsmodelle, die unter den Überschriften „Überwachungskapitalismus“ (Zuboff, 2018) und „Datenkolonialismus“ (Couldry & Mejias, 2019) kritisch diskutiert werden. Möglich werden hierdurch auch vollkommen neue Formen der „Automatisierung“ (Diakopoulos, 2019) von Kommunikation. Der Umstand, dass aktuelle Veränderungen im Bereich von Medien und Kommunikation kaum erfasst werden können, ohne auch die technologische Komponente im Blick zu haben, hat das Verhältnis der Kommunikations- und Medienwissenschaft zu anderen Disziplinen stark verändert. All dies hat eine breite Diskussion im Fach ausgelöst, wie die Kommunikations- und Medienwissenschaft in diesen „datengetriebenen Zeiten“ aussehen kann und soll (Brosius, 2016; Hepp, 2016; Jarren, 2016; Krüger & Meyen, 2018; Strippel et al., 2018; Theis-Berglmair, 2016).

Mit diesen Veränderungen ging eine **Verschiebung der Methoden der Kommunikations- und Medienwissenschaft** einher. Die Daten, die Nutzer*innen von digitalen Medien hinterlassen, bieten für Forscher*innen eine neue Quelle von Information. Diverse Online-Dienste, Plattformen und andere digitale Angebote generieren große Mengen an Daten, auf die Forscher*innen aufbauen können. Gängige Möglichkeiten der Automatisierung bieten neue Chancen für die Auswertung insbesondere großer Datenmengen. Gleichzeitig fordert die zunehmende Komplexität heutiger Medienrepertoires die Forschung heraus, neue qualitative Erhebungs- und Analyseverfahren zu entwickeln, die immer wieder mit digitalen Tools arbeiten. Und auch Studien zur Medien- und Kommunikationsgeschichte können anders angelegt werden, wenn ein Zugriff auf digitale Archive möglich ist. Entsprechende Veränderungen der Methoden der Sozial- und Geisteswissenschaften im Allgemeinen und der Kommunikations- und Medienwissenschaft im Speziellen werden unter Überschriften wie „computational social sciences“ (Lazer et al., 2009), „digital methods“ (Rogers, 2021), „digital social research“ (Veltri, 2020) oder „cultural analytics“ (Manovich, 2020) diskutiert. Diese methodischen Innovationen sind eine wichtige und mit Blick auf den veränderten Begriffs- und Theorieapparat notwendige Bereicherung für die Kommunikations- und Medienwissenschaft. Sie verschieben durch ihren nicht selten explorativen Charakter die Grenzen zwischen quantitativen und qualitativen Verfahren und denen der Theoriefindung und -prüfung. Dabei haben all diese Methoden eines gemein: Software – konkreter: Forschungssoftware – hat eine ganz zentrale Bedeutung für sie (Hepp, Loosen & Hasebrink, 2021).

Folgt man der Deutschen Forschungsgemeinschaft (DFG), so bezeichnet der Ausdruck „Forschungssoftware“ Software „zur Simulation, zur Generierung, Verarbeitung, Analyse und Visualisierung von Forschungsdaten sowie zur Steuerung von Forschungsgeräten und Experimenten“ (Katerbow & Feulner, 2018: 5). Dabei steht *selbst entwickelte* Forschungssoftware in Abgrenzung zur Nutzung von (kommerziellen) Softwareanwendungen (SPSS, MAXQDA, HyperResearch etc.) und in Abgrenzung zur Nutzung von

Infrastruktursoftware bzw. -diensten (Amazon Mechanical Turk, Zenodo, OSF etc.) (zu Infrastruktur siehe Strippel 2021). Die Grenzen sind allerdings fließend, indem Forschungssoftware einerseits selbst entwickelte Anwendungen aus Forschungsprojekten heraus meint, andererseits etwa die Entwicklung von R-Paketen eher Infrastrukturcharakter aufweisen kann.

Seit langem ist Software nicht aus der Forschung der Kommunikations- und Medienwissenschaft wegzudenken: Die quantitative Forschung war und ist beispielsweise eng mit der Verwendung von Software zur Datenauswertung verbunden, die qualitative Forschung etwa mit Kodiersoftware. Computergestützte Methoden sind heute aber deutlich mehr als die Anwendung von (meist kommerzieller) Standardsoftware. Die Art und Menge von Daten, aber auch der Zugang zu selbigen, etwa über entsprechende Schnittstellen (APIs), macht spezielle, im Forschungskontext selbst entwickelte und kontinuierlich angepasste Software nötig, um überhaupt auf entsprechende digitale Daten zugreifen und sie auswerten zu können (Puschmann & Ausserhofer, 2017; Hogan, 2018). Hinzu kommt, dass auch beim qualitativen Vorgehen Software-Entwicklungen an Bedeutung gewinnen, um Daten erheben und auswerten zu können (Hasebrink & Hepp, 2017). Als Kommunikations- und Medienwissenschaft denken wir entsprechend anders als bisher über Forschungssoftware nach, stellen neue Anforderungen an ihre Qualität und erwarten gleichsam neue Kompetenzen des Umgangs mit der Programmierung und der Pflege.

Box 1: Beispiele für Forschungssoftware in der Kommunikations- und Medienwissenschaft

Facepager: Facepager ermöglicht einen Einstieg in die automatisierte Datenerhebung ohne Programmierkenntnisse, zum Beispiel über APIs von YouTube, Twitter, Wikipedia oder von Cloud-Computing-Diensten von Amazon und Google. https://github.com/strohne/Facepager/releases

Quanteda: Ein R-Paket für die Verwaltung, Aufbereitung und (automatisierte) Analyse von Textdokumenten. https://quanteda.io/

Specr: Ein R-Paket für Multiverse Analysen (Spezifikationskurvenanalysen). https://cran.r-project.org/web/packages/specr/index.html

Tosca: Ein R-Framework für die statistische Analyse von Inhaltsanalysen. https://cran.r-project.org/web/packages/tosca/index.html [02.02.2021]

MeTag: Das Programm unterstützt das Führen von Medientagebüchern und besteht aus zwei Komponenten: MeTag Analyze zum Anlegen von Projekten und Analysieren erhobener Daten sowie MeTag als Smartphone-App (Apple und Android), mit Hilfe derer Teilnehmer*innen ein Medientagebuch führen können. https://mesoftware.org/index.php/metag/

Vor dem Hintergrund dieser Veränderungen ist das Ziel des vorliegenden Papiers, den aktuellen Stand der Forschungssoftware in der Kommunikations- und Medienwissenschaft sowie die hierbei bestehenden Herausforderungen und Perspektiven für zukünftige Entwicklungen zu umreißen. Dabei richtet sich dieses Papier an verschiedene Akteur*innen: Erstens an die Fachgesellschaft, zweitens an die Forschungsförderung, drittens an die Instituts- und Universitätsleitungen, viertens an Zeitschriften und Verlage sowie fünftens an uns als Forscher*innen und Dozent*innen selbst. Wir alle sind aufgerufen, gemeinsam Forschungssoftware als ein wichtiges „Tool“ (van Es, Wieringa, & Schäfer, 2021) der Kommunikations- und Medienwissenschaft ernst zu nehmen und kritisch zu reflektieren, wenn wir das Fach im „digitalen Wandel“ angemessen positionieren wollen.

2. Forschungssoftware zwischen Nutzung, Anpassung und Entwicklung

Unabhängig davon, ob es um ein Forschungsprojekt geht oder den Einsatz von Forschungssoftware in der Lehre – prinzipiell lassen sich a) die Nutzung bestehender Lösungen, b) die Anpassung und c) die Neuentwicklung von Forschungssoftware unterscheiden. Wir wollen mit dem Papier dazu anregen, diesen „Dreischritt" als die normale Abfolge der Annäherung an Forschungssoftware anzusehen: Entwicklung und Aufrechterhaltung von Forschungssoftware ist ressourcenintensiv, weshalb vor der Neuentwicklung der Blick auf das Bestehende und die Möglichkeit von dessen Anpassung stehen. Dabei gilt es, verschiedene Fragen im Blick zu haben:

- *Zielsetzung:* Ist die Forschungssoftware reines Hilfsmittel zur Bearbeitung der Forschungsfragen? Oder ist die Entwicklung der Forschungssoftware der eigentliche Gegenstand des Projekts?
- *Kompetenzen:* Wer im Team hat welche Kompetenzen? Will man die technologische und organisatorische Umsetzung im Team realisieren oder ist es sinnvoller, externe Kompetenzen hinzuzuziehen?
- *Anforderungen:* Geht es um eine einmalige Lösung für ein spezifisches Forschungsprojekt? Oder hilft die Forschungssoftware bei Aufgaben, die auch in anderen Projekten auftreten (können) und ist sie deshalb für viele von Interesse?

Nutzung bestehender Software

Dient Forschungssoftware der Bearbeitung bestimmter Forschungsfragen, ist also nicht die Entwicklung der Software Kern des Projekts, liegt es nahe, auf bestehende Lösungen zurückzugreifen. Die **Nutzung bestehender Forschungssoftware** spart den Beteiligten Zeit und Entwicklungsressourcen. Allerdings wird dies mit einer geringeren Flexibilität im Forschungsprozess und entsprechenden Ausgaben erkauft. Für die Wahl der Software bieten sich zunächst Repositorien, Software-Verzeichnisse und einschlägige Fachzeitschriften zur Recherche an, wobei neben dem eigentlichen Einsatzzweck insbesondere Kriterien der Aktualität, der rechtlichen Rahmensetzung sowie das Vorhandensein einer hinreichenden Gruppe von Nutzer*innen der Forschungssoftware berücksichtigt werden sollten (für einen Überblick wichtiger Verzeichnisse siehe Box 2).

Box 2: Informationsquellen für kommunikations- und medienwissenschaftliche Forschungssoftware

AoIR-Mailingliste: Auf der Mailingliste der Association of Internet Researchers wird kontinuierlich Forschungssoftware vorgestellt und es werden Erfahrungen mit dieser und deren Entwicklung ausgetauscht. https://aoir.org

Data Collection Tools-Liste des HBI Social Media Observatory: Übersicht hilfreicher Tools für die Sammlung digitaler Daten. https://smo-wiki.leibniz-hbi.de/

de-RSE (Research Software Engineers): Fächerübergreifende Gemeinschaft von Forschungssoftware-Entwickler*innen inklusive Mailingliste. https://de-rse.org/

DGPuK-Webseite: Auf der Webseite der DGPuK befindet sich eine Datenbank, die einen Überblick über Forschungssoftware in der Kommunikations- und Medienwissenschaft gibt. https://www.dgpuk.de/de/forschungssoftware.html

Digital Methods Initiative: Liste der verschiedenen Tools der Digital Methods Initiative, Universität Amsterdam. https://wiki.digitalmethods.net/Dmi/ToolDatabase

Viele Tools sind im Fach in Gruppen von Entwickler*innen und Nutzer*innen entstanden. Gerade wenn man sich an einzelne Arten von Forschungssoftware neu annähert, lohnt es sich Kontakt aufzunehmen. Es geht dabei nicht nur um Wissenstransfer, sondern die Beteiligung an solchen Gruppen kann auch ein

Beitrag zur Aufrechterhaltung und Weiterentwicklung der jeweiligen Forschungssoftware sein und die langfristige Qualitätssicherung fördern. Nutzer*innen bestehender Forschungssoftware sollten – wie bei anderen wissenschaftlichen Beiträgen auch – die Entwickler*innen zitieren, Rezensionen verfassen, Erfahrungsberichte und Empfehlungen zur Community beisteuern. Vorhandene Strukturen des Informationsaustausches und der Vernetzung sollten genutzt und in Forschungsprojekte sowie -ausschreibungen integriert werden. Begutachtung von Forschungssoftware in Fachzeitschriften, Treffen von Nutzer*innen (ggf. mit Entwickler*innen) im Rahmen von Tagungen sowie Finanzierungsoptionen für das Community-Management rund um Forschungssoftware könnten die Nutzung bestehender Software attraktiver und nachhaltiger machen.

Anpassung vorhandener Software

Erfüllt keine bestehende Softwarelösung die Anforderungen eines Forschungsprojekts, bietet sich mitunter die **Anpassung vorhandener Software** an. Damit ist gemeint, dass Forschungssoftware so modifiziert wird, dass sie den jeweiligen speziellen Bedürfnissen in der Forschung und Lehre entspricht. Die Spanne reicht dabei vom Einbetten bereits publizierter Software in den eigenen Arbeitsprozess über das aktive Weiterentwickeln der Software (beispielsweise bei einem R-Paket) bis hin zum Ergänzen größerer Code-Bestandteile (wobei die rechtlichen Rahmenbedingungen zu beachten sind). All dies setzt die Fähigkeit voraus, sich in bestehenden Code einzuarbeiten und diesen selbst zu entwickeln. Dabei ist der Aufwand sowohl für die Maintainer – die Hauptentwickler*innen, die für ein Entwicklungsprojekt die Entscheidungen treffen – als auch für „Adaptierende“ nicht zu unterschätzen und sollte entsprechend eingeplant werden. Über die reine Nutzung hinaus ermöglicht die Anpassung vorhandener Software deutlich mehr **Flexibilität** im Forschungsprozess. Dabei bleibt bei gleichzeitiger Einhaltung bestehender Qualitätsstandards der eigene Einsatz moderat.

Zusätzlich trägt die Adaption zum Aufbau einer vielfältigen Landschaft von Forschungssoftware bei, da sie die **kontinuierliche Weiterentwicklung bestehender Forschungssoftware** in einer modularen Struktur fördert und so zu verhindern hilft, dass isolierte Einzelentwicklungen etwa aus Zeitmangel eingestellt werden (müssen) (vgl. von Nordheim et al., 2021). Für eine angemessene Qualitätssicherung ist es wichtig, sich bereits verfügbaren Modulen und deren Entwickler*innen anzuschließen sowie „style guides“ und „good practices“ der zu adaptierenden Software zu beachten (vgl. Martin, 2008). Die Software selbst muss dafür in ihrer Lizenz eine Adaption zulassen, gut dokumentiert und in ihren Schnittstellen zuverlässig sein, bestenfalls sichergestellt über entsprechende Testszenarien. Im Idealfall gibt es auch Ansprechpartner*innen, die einer Anpassung offen gegenüberstehen und helfen können.

An die Adaption von Forschungssoftware werden konkrete **Anforderungen** gestellt: Es gilt die angepassten Teile der bestehenden Software adäquat zu lizenzieren, zu dokumentieren und zu testen, um dann deren Weiterentwicklung wiederum zu ermöglichen. Die Voraussetzung einer zielführenden Anpassung ist also eine entsprechende Qualitätsorientierung nicht nur der Hauptentwickler*innen, sondern auch der mitarbeitenden Entwickler*innen. Damit die gemeinsame Entwicklung gelingen kann, ist eine institutionalisierte Entwicklungsinfrastruktur mit Versionsverwaltung, Testsystemen (CI/CD) und Hosting-Möglichkeiten wünschenswert, die auf dem Stand der Technik gehalten wird und über einzelne Einrichtungen hinaus für Kooperationen verfügbar sein sollte (bspw. über GitHub).

Vollständige Software-Neuentwicklung

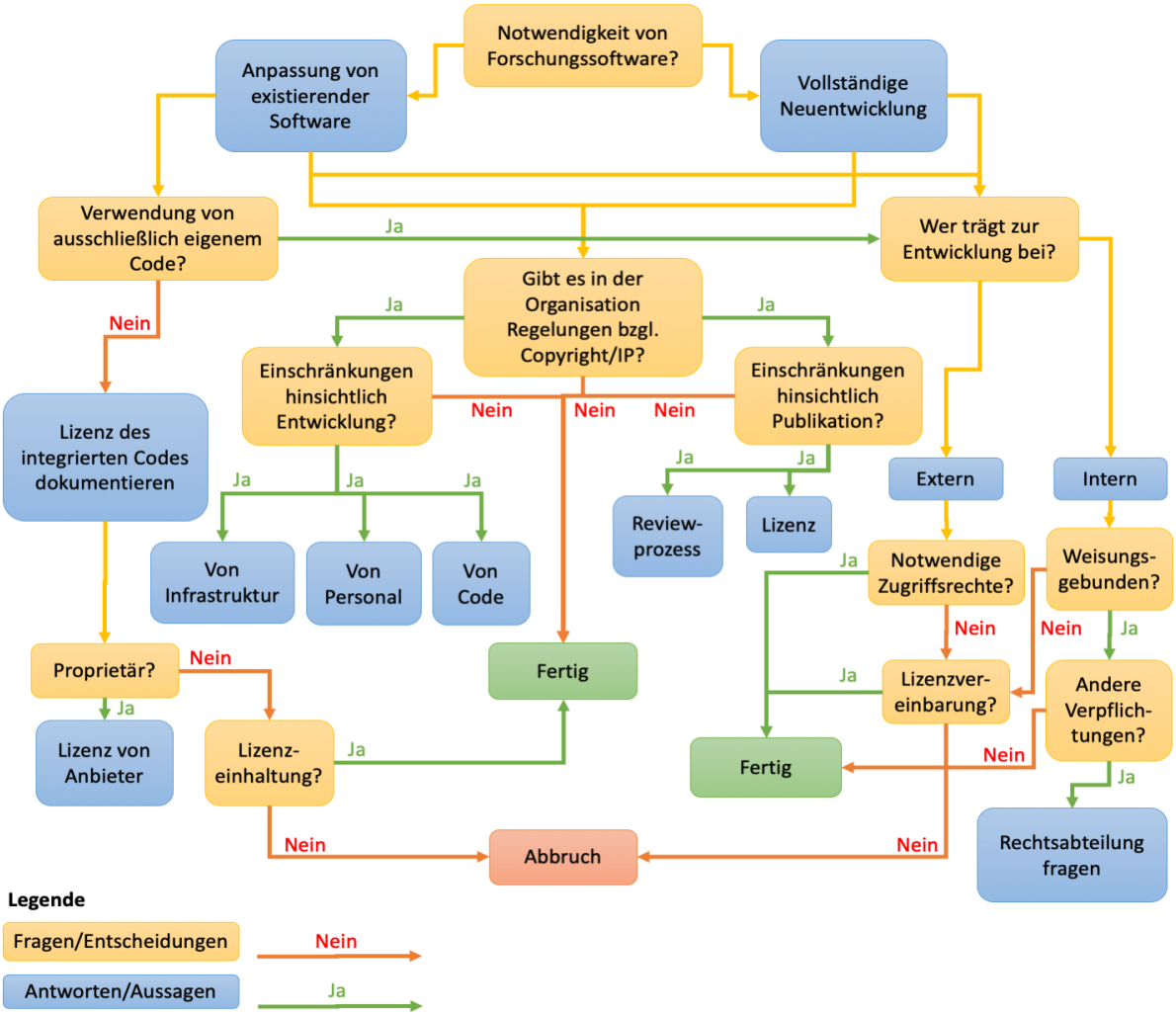
Die größte Freiheit mit Blick auf den Forschungszweck ermöglicht die **komplette Neuentwicklung** von Forschungssoftware. Dies kann in unterschiedlichen Abstufungen geschehen, etwa im Rahmen schneller und flexibler eigener Entwicklung, nachhaltiger und zielgerichteter professioneller Entwicklung im Team oder eingekaufter Expertise durch entsprechende Dienstleister (Storer, 2017). Doch wenngleich sich nahezu alle Projekte inhaltlich für eine komplette Neuentwicklung eignen, setzen die damit verbundenen Anforderungen klare Grenzen: Neben einem erhöhten zeitlichen Aufwand sind vertiefte Kenntnisse der Softwareentwicklung und des Projektmanagements unerlässlich; sind diese nicht vorhanden oder können nicht eingekauft werden, ist eine zeit- und kostenintensive Einarbeitung notwendig. Diese gilt es immer abzuwägen, auch im Hinblick auf die Möglichkeiten und Beschränkungen der Qualifikation von Wissenschaftler*innen, vor allem Doktorand*innen. Inwieweit eröffnen sich an dieser Stelle Qualifikationschancen? Wo wird die Beteiligung an der Entwicklung von Forschungssoftware zu einer Zusatzaufgabe, die die eigene fachwissenschaftliche Qualifikation behindert? Spielt man mit dem Gedanken einer Neuentwicklung, sind vielfältige Entscheidungen zu treffen, die Einzelne schnell überfordern können (siehe Abbildung 1). Wo innerhalb der Universitäten und Forschungseinrichtungen einschlägige Beratungsstellen vorhanden sind, sollten sie hinzugezogen werden.

Um die **Nachhaltigkeit kompletter Neuentwicklungen** sicherzustellen, sollte bereits bei der Planung konsequent auf Zusammenarbeit und offene Lizenzierung gesetzt werden. Außerdem sollte berücksichtigt werden, dass sich das Wissen nicht bei einzelnen Personen kumuliert und dass die Weitergabe des Wissens sichergestellt wird. Auch das spätere Einholen von Lizenzen ist um ein Vielfaches schwieriger als eine konsequente offene Lizenzierung von Beginn an (siehe Box 3). Auf eine solche offene Lizenzierung sollten auch Werk- und Dienstverträge und die Urheberrechtserklärungen in diesen ausgerichtet sein, nicht zuletzt um spätere Streitigkeiten zu vermeiden.

Box 3: Lizenzen von Forschungssoftware

Die Wahl der Lizenz einer Forschungssoftware hängt vom konkreten Projektvorhaben ab. Wir empfehlen grundsätzlich die Nutzung offener Lizenzen (z.B. MIT, https://opensource.org/licenses/mit-license.php), die nicht nur gute Bedingungen für kollaborative (Weiter-)Entwicklung schaffen, sondern auch den (akademischen) Entwickler*innen Sicherheit geben, den Zugang zur Software zu bewahren (z.B. im Falle eines Universitätswechsels). Die Wahl der richtigen Lizenz für wissenschaftliche Zwecke ist eine bislang kaum beachtete Problemstellung (Vendome et al., 2018), die zahlreiche nicht antizipierte Folgekomplikationen nach sich ziehen kann. Neben der rechtlichen Komplexität erschwert die große Zahl von über 100 gängigen Open-Source-Lizenzen die Entscheidung (https://www.gnu.org/licenses/license-list.html, https://opensource.org/licenses/category). Nach dem derzeitigen Stand ist in jedem Fall die Konsultation von Expert*innen in dieser Rechtsmaterie empfehlenswert (Anzt et al., 2020). Grundsätzlich sollte hervorgehoben werden, dass möglichst offene Lizenzen dem Geist der aufgeklärten Wissenschaft am ehesten entsprechen, auch wenn in Einzelfällen (z.B. aufgrund vertraglicher Verpflichtungen) weniger freie Lizenzen gewählt werden müssen.

Abbildung 1: Aspekte der Entwicklung von Forschungssoftware



Bei der Neuentwicklung von Forschungssoftware müssen zahlreiche rechtliche Aspekte vorab geplant werden. Dazu gehören Lizenzierung und Nutzung durch andere (inklusive Kompatibilität mit Open-Science-Grundsätzen), langfristige Verpflichtungen und Haftungsrisiken, Datenschutz sowie Klärung der Urheberschaft (individueller Entwicklung, interdisziplinärer Teams, ausgegründeter Firmen oder Arbeitgeber). Einen guten Überblick geben hier die Entscheidungsbäume in dem Papier „An environment for sustainable research software in Germany and beyond: current state, open challenges, and call for action“, dessen jeweils aktuelle Version unter dem Link https://f1000research.com/articles/9-295/v1 geladen werden kann.

Quelle: Eigene Abbildung basierend auf Entscheidungsbäumen in Struck et al., 2020

Je nach Größe des Projekts kann die Zusammenarbeit verschiedene Ausmaße annehmen. Generell ist es aber sinnvoll, nicht nur an die eigenen Interessen bei einer Neuentwicklung zu denken, sondern auch an andere mögliche Nutzer*innen. Dies fördert den Aufbau einer **Community von zukünftigen Nutzer*innen und Entwickler*innen**, die in vielen Fällen die Voraussetzung für eine dauerhafte Verfügbarkeit der Software ist. Der Einbezug anderer kann dabei unterschiedliche Abstufungen haben. Als Minimum sollten

über die Fachgemeinschaft mögliche Nutzer*innen gesucht und bei der Konzeption von Features der Software einbezogen werden. Damit potenzielle Diskrepanzen zwischen zuvor vorgestellten und bestehenden Anwendungsszenarien sichtbar werden, kann dies auch kleinere empirische Studien einbeziehen, in denen die Entwickler*innen unvoreingenommene Rückmeldungen von Nutzer*innen einholen (Sharp, Preece, & Rogers, 2019). Das Maximum sind partizipative Entwicklungsprojekte (Simonsen & Robertson, 2013), die – wie beispielsweise bei Co-Creation (Sanders & Stappers, 2008, speziell für Forschungssoftware in der Kommunikations- und Medienwissenschaft: Hohmann, 2021) – Anwender*innen umfassend in die Konzeption und Entwicklung von Forschungssoftware einbeziehen. Bei mittelgroßen Projekten kann es zusätzlich sinnvoll sein, Kooperationen mit anderen Instituten oder Disziplinen zu suchen. Für solche Kooperationen sind die Organisation von Arbeitsgruppen auf Ebene der Fachgesellschaft oder Vernetzungsformate bei Tagungen (Werkstattberichte, Panels, World Cafés etc.) hilfreich. Bei einem großen Forschungssoftware-Projekt sollte zudem geprüft werden, ob es sich um eine Aufgabe handelt, die als Infrastruktur von höherer Stelle (Fachgesellschaft, zentraler Forschungseinrichtung) dauerhaft geleistet werden sollte.

Wie unsere Darlegungen in diesem Abschnitt gezeigt haben, geht es aktuell bei der Forschungssoftware in der Kommunikations- und Medienwissenschaft um ein breites Spektrum von Nutzung, Anpassung und Entwicklung. Alle drei Zugänge stehen dabei in enger Wechselbeziehung, da nur im Zusammenspiel eine **vielfältige und nachhaltige Landschaft von Forschungssoftware** entsteht. Um dies zu fördern, müssen entsprechende Voraussetzungen erfüllt sein:

- Die **Entwickler*innen und Forscher*innen** müssen klar auf Open-Science-Prinzipien setzen, da nur so eine vielfältige und nachhaltige Entwicklung von Forschungssoftware möglich ist.
- Die **Wissenschaftsförderung** (DFG, BMBF etc.) ist gefordert, mehr Förderformate und Richtlinien zu entwickeln, die eine nachhaltige und zugleich vielfältige (Weiter-)Entwicklung von Forschungssoftware ermöglichen und nachhaltig sicherstellen.
- Im **Wissenschaftsmanagement** (Leitung von Universitäten und Forschungseinrichtungen) sollten Ansprechpartner*innen und zentrale Stellen für die Belange von Forschungssoftware eingerichtet werden. Zudem sollen Verträge so gestaltet sein, dass sie quelloffene Software mindestens ermöglichen, sollte eine moderne und adäquate Entwicklungsinfrastruktur bereitgestellt werden, und es sollten entfristete Stellen geschaffen werden, um die Kontinuität von Entwicklungen sicherzustellen.
- **Fachzeitschriften** und **Verlage** sollten Autor*innenrichtlinien für den Umgang mit Forschungssoftware und deren Verwendung innerhalb veröffentlichter Beiträge erarbeiten.
- Die **Fachgesellschaft** sollte Austauschprozesse über Forschungssoftware unterstützen und die sich daraus ergebenden Forderungen wissenschaftspolitisch vertreten.

3 Vielfalt und Standardisierung

Wie der letzte Abschnitt deutlich gemacht hat, ist die Entwicklung von Forschungssoftware in einem **Spannungsverhältnis von Vielfalt und Standardisierung** zu sehen: Auf der einen Seite lebt Forschungssoftware von der Vielfalt, geht es doch darum, Lösungen für sehr spezifische Forschungsfragen zu finden, die gerade nicht mit Standard-Applikationen bearbeitet werden können. Die Entwicklung von Forschungssoftware ist eingebettet in den wissenschaftlichen Erkenntnisprozess und erfordert somit ein hohes Maß an Flexibilität. Anforderungen an die Software sind so teilweise vorab unbekannt oder nur vage benennbar und verändern sich möglicherweise im Forschungsprozess. Der Formalisierung des Entwicklungsprozesses von

Forschungssoftware sind daher Grenzen gesetzt (Johanson & Hasselbring, 2018). Auf der anderen Seite ist eine gemeinsame, nachhaltige und kooperative Entwicklung von Forschungssoftware nur dann möglich, wenn gewisse Standards eingehalten werden. So ist es möglich, dass beispielsweise einzelne Module von phasenweise mitarbeitenden Entwickler*innen realisiert werden, bzw. dass generell umfassende Kooperationsstrukturen entstehen. Eine bestimmte Standardisierung erscheint also zielführend. Das Spannungsverhältnis von Vielfalt und Standardisierung ist somit stets Teil der Entwicklung und Nutzung von Forschungssoftware. Es kann nicht aufgelöst werden, aber über eine entsprechende Dokumentation, zielführende Dateiformate und eine Austauschkultur sehr produktiv für die fortlaufende Weiterentwicklung von Forschungssoftware sein.

Dokumentation

In Ergänzung zu Handbüchern für Nutzer*innen stellt die Code-Dokumentation einen entscheidenden **externen Erfolgsfaktor** für Forschungssoftware dar, denn sie erleichtert potenziellen Anwender*innen die (fehlerfreie) Nutzung, ermöglicht konstruktives Feedback (z.B. in Form von Bug Reports) und senkt (gemeinsam mit klar strukturiertem Code) die Einarbeitungszeit für neue Entwickler*innen. Daneben kann eine erfolgreiche Dokumentation auch **projektintern** hilfreich sein, indem sie zwischen Rollen (Programmierer*innen / Anwender*innen / Projektleiter*innen) vermittelt, eigene Standards definiert und – neben Spezifikationen – eine gemeinsame Zielorientierung unterstützt.

Allerdings ist die Erstellung von Dokumentation zumeist nicht im primären Fokus der Entwickler*innen: Sie erfordert Zeit und Mühe, die oft an anderer Stelle – für die „eigentliche" Programmierung – gebraucht wird. Zudem muss sie laufend aktualisiert werden, um ihren Zweck zu erfüllen. Der Nutzen von Dokumentation besteht also insbesondere in **langfristigen Zielsetzungen**, der Nachvollziehbarkeit der Funktionalitäten und der Möglichkeit, dass sich andere an dem Entwicklungsprozess beteiligen.

Für eine erfolgreiche **Integration der Dokumentation in den Entwicklungsprozess** von Forschungssoftware sollte daher der mittel- und langfristige Nutzen anerkannt und fest in der Projektplanung und -durchführung verankert werden. Gleichzeitig ist es wünschenswert, den Mehraufwand einer guten Dokumentation durch Automatisierung so weit wie möglich zu begrenzen: Wenn Dokumentation etwa halbautomatisch erstellt wird (z.B. aus Docstrings im Code, vgl. Doxygen), entfällt ein erheblicher Teil des manuellen Aufwands bei der Pflege. Für eine transparente und nachvollziehbare Dokumentation empfehlen wir deswegen eine Orientierung an entsprechenden Style Guides, die Nutzung verbreiteter Entwicklungsumgebungen und Publikationsplattformen (siehe Box 4 mit Beispielen für R und Python).

Box 4: Ressourcensammlung für den Entwicklungsprozess am Beispiel von R und Python

- Style Guides
  - R: Tidyverse style guide
  - Python: PEP 8
- Entwicklungsumgebungen
  - R: RStudio
  - Python: PyCharm (kostenlose Pro-Lizenz für akademische User)
- Publikationsplattformen
  - GitHub (kostenloser Teams-Account für akademische User)
  - GitLab (kostenloser Gold-Account für akademische User)

Als ein Teilaspekt der Dokumentation sollte auch die **Publikation von Forschungssoftware** angesehen werden, einschließlich ihrer allgemein zugänglichen Speicherung und Archivierung. Diese Publikation dient zwei Zielen: Erstens werden im Hinblick auf die gegenwärtige Nutzung die Transparenz, Nachvollziehbarkeit und Replizierbarkeit der Forschung mittels der entsprechenden Software sichergestellt. Zweitens werden im Hinblick auf zukünftige Forschung der archivarischen Pflicht von Wissenschaft Rechnung getragen und die dauerhafte Verfügbarkeit der Forschungssoftware gesichert.

Ein Grundgerüst für die Dokumentation sind **Versionskontrollsysteme** wie Git oder Mercurial, die in Kombination mit **öffentlichen Repositorien** (z.B. auf GitHub oder GitLab) ein feingliedriges Abbild des Codes und seiner Entwicklung sicherstellen. Sie sind gerade für kollaborative Projekte hilfreich, da sie die gemeinsame Dokumentation von Problemen und Bugs ermöglichen. Gute Entwicklung lebt in diesem Sinne von einem „Open-Source-Bewusstsein“ und der Bereitschaft zum Zur-Verfügung-Stellen von Software. Da sich die Systeme und Repositorien ständig ändern können, sind unsere Empfehlungen für die Speicherung von Forschungssoftware dynamisch zu sehen. Kommerzielle Plattformen (wie die oben genannten) bieten den Vorteil eines weit verbreiteten Quasi-Standards. Allerdings unterliegen sie potenziell rechtlichen Veränderungen, und Entwickler*innen können so zur Löschung von Code und Daten gezwungen werden. Kommerzielle Plattformen sind daher nur in Kombination mit zusätzlichen archivarischen Kopien geeignet.

Forschungssoftware ermöglicht vielfältige Forschungen und ermächtigt Forscher*innen. Im Falle von Fehlern kann sie aber zu weit verbreiteten und gegebenenfalls systematischen Irrtümern führen (Ziemann, Eren, & El-Osta, 2016). Eine reine Dokumentation und Veröffentlichung von Quellcode ist die Voraussetzung für eine effektive **Qualitätskontrolle**. Dokumentation und Veröffentlichung alleine können aber zu kurz greifen, da sie keine qualifizierte und tiefgreifende Beschäftigung mit dem Code garantieren. Solch eine intensive Fehlersuche findet in der Softwareentwicklung in sogenannten **Code Reviews** statt. Der damit verbundene erhebliche Aufwand legt nahe, für solche intensiven Prüfungen eigene Motivationsstrukturen zu schaffen. In der Informatik existiert beispielsweise bereits der „Most Reproducible Paper Award“ der SIGMOD (Special Interest Group on Management of Data der Association for Computing Machinery) (Hasselbring et al., 2020). Ähnliche Würdigungen erscheinen an dieser Stelle gegebenenfalls auch für die Kommunikations- und Medienwissenschaft sinnvoll.

Dateiformate

Im Spannungsverhältnis von Vielfalt und Standardisierung sind auch die Dateiformate zu sehen, mittels derer eine Forschungssoftware arbeitet. Die **Vielfalt von Dateiformaten** ermöglicht eine Berücksichtigung sehr spezifischer Forschungsinteressen. Es gibt eine große Zahl an Datenformaten und -typen, sowohl bei Inputdaten von Forschungssoftware als auch bei Outputdaten (z.B. tabellarische Daten, Markup-Daten, hierarchische Daten, Multimedia-Daten). Auch wegen der notwendigen Pluralität der Forschungsinteressen erscheint es uns nicht sinnvoll, Empfehlungen für ein einzelnes Format zu geben. Um Entwicklung besser und nachhaltiger zu gestalten sowie den Transfer zwischen unterschiedlichen Datenformaten (und somit auch zwischen einzelnen Softwarelösungen) zu vereinfachen, erscheint es uns aber wichtig, einzelne **grundlegende Anforderungen an Datentypen und -formate** zu formulieren: Um einen größtmöglichen Nutzen von Forschungssoftware für möglichst viele Forscher*innen sicherzustellen, empfehlen wir, wo immer möglich offene, etablierte, standardisierte und für andere Forscher*innen am besten anschlussfähige Formate zu verwenden. Dies ermöglicht auch die direkte Lesbarkeit von Daten in gängigen Repositorien (z.B. GitHub, OSF), die zugleich eine Orientierungsmöglichkeit über verbreitete Formate bieten.

Der **Datenimport und Datenexport** sollte nach Möglichkeit über etablierte Module, Bibliotheken und Pakete erfolgen (z.B. json- und csv-Module in Python anstelle eigener Entwicklung), um Anschlussfähigkeit herzustellen und auch Grenzfälle abzudecken. Es empfiehlt sich außerdem, etablierte Standards für bestimmte Datentypen anzuwenden (z.B. „tidy data“ für tabellarische Daten). Hilfreich kann es je nach Ziel und Größe der Projekte wiederum sein, andere Forscher*innen frühzeitig in den Entwicklungsprozess einzubinden, um so zu klären, welche Outputformate für die Weiterverarbeitung der Daten sinnvoll sind (siehe unsere Hinweise in Abschnitt 2).

Box 5: Gängige Formate für den Datenaustausch

Comma Separated Values (CSV): Für tabellarische Daten eignen sich CSV-Dateien. In der ersten Zeile stehen in der Regel die Spaltenbezeichnungen, jeder Datensatz wird in einer eigenen Zeile erfasst, die Werte werden zum Beispiel durch Kommata oder Semikola voneinander getrennt.

Javascript Object Notation (JSON): Das JSON-Format bildet komplexe Datenstrukturen ab und ist dabei sowohl menschen- als auch maschinenlesbar. Es kommt häufig im Zusammenhang mit APIs zum Einsatz. Die Daten werden als Name-Wert-Paare erfasst.

Extended Markup Language (XML): Ähnlich wie das im Web gebräuchliche HTML, bietet das XML-Format eine hierarchische Strukturierung von Daten. Die Daten werden durch verschachtelte XML-Tags erfasst. Eine Standardisierung erfolgt durch XML-Schemata.

Zusätzlich ist zu bedenken, dass die Bedeutung der Werte und der Aufbau der Dokumente dokumentiert werden sollten. Hierfür bilden sich bereichsspezifische Konventionen und Standards aus, die möglichst beachtet werden sollten. Ein Beispiel für die Spaltenbezeichnungen von Textkorpora findet sich unter https://github.com/ropensci/tif. Vorschläge zur Dokumentation von großen Datensätzen bieten Gebru et al. (2020). Um Machine-Learning-Modelle nachvollziehbar zu beschreiben, haben Mitchell et al. (2019) Hinweise zusammengetragen.

Austauschkultur

Um einen Austausch über die Entwicklung von Forschungssoftware im Hinblick auf deren notwendige Dokumentation und die zugrundeliegenden Dateiformate sicherzustellen, erscheint es uns notwendig, in der Kommunikations- und Medienwissenschaft einen **dauerhaften gemeinsamen Austausch** in und über verschiedene Entwickler*- und Nutzer*innengruppen hinweg zu etablieren. Als sinnvolle Veranstaltungen erscheinen uns regelmäßige Software-spezifische Workshops bei Tagungen, ob als partizipative Anwendungsworkshops (zum Austausch über den Umgang mit einer Software) oder als Co-Creation-Workshops (als Basis für zukünftige Entwicklungen). Solche Angebote müssen nicht von Entwickler*innen angeleitet werden, sondern können auch von Software-Anwender*innen getragen werden. Neben dem Erfahrungsaustausch bieten solche Formate die Möglichkeit, die (Weiter-)Entwicklung von Forschungssoftware und ihre finanzielle Unterstützung zu sondieren. Über solche Einzeltreffen hinweg geht es darum, einen kontinuierlichen Austausch über Forschungssoftware in der Fachgesellschaft zu etablieren, um Anforderungen an Qualität, Dokumentation und Dateiformate fortlaufend auszuhandeln.

Das Spannungsverhältnis von Vielfalt und Standardisierung verweist damit aus unserer Sicht auf verschiedene Anforderungen:

- Eine wichtige Anforderung an **Entwickler*innen und Forscher*innen** ist es, in einer „Atmosphäre der Dokumentation“ Versionskontrollen einzusetzen, transparent und ggf. automatisiert zu dokumentieren sowie Formate zu verwenden, die für Datenaustausch und langfristige Datensicherung geeignet sind.
- Die **Wissenschaftsförderung** sollte bei der finanziellen Unterstützung von Projekten Wert auf deren Dokumentation und die Kompatibilität von Dateiformaten legen und hierfür die notwendigen Ressourcen zusätzlich zur Verfügung stellen.
- Für **Fachzeitschriften** gilt, dass neue (digitale) Formate für Publikationen mit Forschungssoftware geschaffen werden sollten und die Dokumentation von Forschungssoftware im Review-Prozess angemessen berücksichtigt werden sollte. Insbesondere empfehlen wir zusätzlich zu thematischen Aufsätzen regelmäßig Softwarebesprechungen zu veröffentlichen (ähnlich wie Buchbesprechungen).
- Die **Fachgesellschaft** sollte den Austauschprozess zwischen Entwickler*innen und Nutzer*innen fördern sowie ein Forum für die Diskussion um geteilte Standards schaffen. Hierbei erscheint uns eine Kooperation mit anderen Fachgesellschaften sinnvoll.
- In der **Lehre und Qualifikation** sollten Kompetenzen vermittelt werden, um Forschungssoftware einschätzen und bewerten zu können, gerade auch im Hinblick auf die Problematik von Vielfalt und Standardisierung. Idealerweise wird dadurch auch eine Grundlage dafür geschaffen, dass sich Wissenschaftler*innen in Qualifikationsphasen später in die Entwicklungsprozesse bestehender Software einbringen können.

4. Finanzierung und Anerkennung

Wie unsere bisherigen Darlegungen mehrfach haben anklingen lassen, ist die Nutzung und Entwicklung von Forschungssoftware an das Vorhandensein entsprechender Ressourcen gebunden. Hierbei unterscheidet sich Forschungssoftware klar von kommerzieller Software: Letztere wird von Unternehmen, die sich für die Weiterentwicklung verantwortlich zeichnen, gegen entsprechende (Lizenz-)Gebühren zur Verfügung gestellt, zunehmend über Abonnementmodelle. Forschungssoftware hat typischerweise einen wesentlich höheren Spezialisierungsgrad und damit einhergehend auch einen wesentlich kleineren Kreis von Nutzer*innen. Entsprechend geht es an dieser Stelle um andere **Modelle der Ressourcensicherung**. Zentral erscheint uns dabei eine angemessene Finanzierung von Forschungssoftware und eine Anerkennung der Leistung ihrer Entwicklung und Bereitstellung.

Finanzierung von Forschungssoftware

Die Entwicklung und fortlaufende Bereitstellung von Forschungssoftware will finanziert werden. Aufgrund ihres bereits genannten Spezialisierungsgrads ist nur in wenigen Fällen die (teilweise) Kommerzialisierung der Software ein gangbarer Weg, ob in Form eines erwerbspflichtigen Kaufprodukts (wie bei SPSS oder MAXQDA) oder in Form der Kommerzialisierung bestimmter Dienstleistungen, Features und Anwendungsbereiche (wie bei RStudio oder SoSciSurvey). Hinzu kommt, dass eine Kommerzialisierung der oben umrissenen Offenheit und Transparenz von Forschungssoftware zuwiderlaufen kann. Besonders wichtig für die Entwicklung und Bereitstellung akademischer Forschungssoftware sind entsprechend **Drittmittelgeber** (insbesondere BMBF, DFG, EU). Hierbei bestehen prinzipiell drei Möglichkeiten der Förderung: Erstens in der Form, dass die Kosten für die Nutzung bestehender Forschungssoftware im Projekt abgerechnet werden können. Zweitens in der Form der Förderung eines bestimmten inhaltlichen Projekts, als dessen

Teil die Forschungssoftware angepasst oder neu entwickelt wird. Drittens in der Form der Förderung der Entwicklung einer bestimmten Forschungssoftware als solcher. Dieser dritte Fall kommt allerdings weit seltener vor, da es nur sehr wenige für die Kommunikations- und Medienwissenschaft anschlussfähige Förderprogramme zur reinen Entwicklung von Forschungssoftware gibt.

Die aktuell übliche Entwicklung von Forschungssoftware über Projektförderungen führt ein grundlegendes **Dilemma der Finanzierung** vor Augen: Erfolgt die Finanzierung über Drittmittel und insbesondere in solchen Projekten, denen eine fokussierte inhaltliche Fragestellung zugrunde liegt, sind die Möglichkeiten einer dauerhaften Programmierung und Dokumentation von Forschungssoftware limitiert. Einzelne Entwicklungen beschränken sich daher oft auf die Laufzeit von Projekten und bestehen selten darüber hinaus.

An dieser Stelle sind sowohl die Universitäten und Forschungseinrichtungen gefragt als auch die Fachgemeinschaft. Universitäten und Forschungseinrichtungen sollten gerade auch im Bereich der Kommunikations- und Medienforschung die kontinuierliche (Weiter-)Entwicklung wichtiger Forschungssoftware als einen Teil ihrer **Infrastrukturleistungen** begreifen und entsprechende Ressourcen, insbesondere in Form von Personal, zur Verfügung stellen. Viele zukünftige innovative Forschungen im Bereich der Kommunikations- und Medienwissenschaft werden dies voraussetzen. Drittmittelstellen der Universitäten sollten sich mit Forschungssoftware auseinandersetzen und Mittel zur Antragsvorbereitung einschließlich der Entwicklung von Prototypen (als Anschubfinanzierung) zur Verfügung stellen.

Die Fachgemeinschaft ist in dem Sinne gefragt, solche Infrastrukturleistungen als gemeinsame Aufgabe anzuerkennen: Bei wichtiger Forschungssoftware erscheint es uns zielführend, eine **Kombination von Finanzierungsmodellen** zu entwickeln, um so die Kosten auf mehrere Schultern zu verteilen. Hierbei bestehen verschiedene Möglichkeiten, die sich entlang unserer Unterscheidung von Nutzung, Anpassung und Entwicklung von Forschungssoftware systematisieren lassen (siehe Box 6, der unsere Unterscheidung aus Abschnitt 2 zugrunde liegt).

Box 6: Mögliche Modelle zur Finanzierung von Forschungssoftware

Modell 1 Nutzung – „Berücksichtigung der Nutzung von Forschungssoftware im Projektantrag“: Bei der reinen Nutzung von Forschungssoftware, die durch andere Institutionen entwickelt wurde, können diese Kosten in eigenen Projektanträgen berücksichtigt werden, was eine Kostenbeteiligung bei den betreffenden Institutionen ermöglicht. Der Betrag im eigenen Projektantrag ist dabei häufig gering; verschiedene solcher „kleineren“ Beträge können aber eine nachhaltige Bereitstellung wichtiger Forschungssoftware für die Scientific Community sicherstellen.

Modell 2 Anpassung – „Forschungsprojekt als Beitrag zur kollektiven Softwareentwicklung“: Will man selbst bestehende Forschungssoftware für ein bestimmtes Forschungsvorhaben anpassen, sollten diese Entwicklungskosten im jeweiligen Projektantrag berücksichtigt werden. Dabei sollte die Entwicklungsleistung so angelegt sein, dass damit ein Beitrag zur nachhaltigen Weiterentwicklung der jeweiligen Forschungssoftware geleistet wird, beispielsweise indem durch die Anpassung für das Eigenprojekt ein zusätzliches Modul der Scientific Community insgesamt beigesteuert wird.

Modell 3 Entwicklung – „institutionell abgesicherte Förderung von Entwicklungsprojekten“: Geht es um die Entwicklung einer neuen Forschungssoftware, besteht ein Modell darin, sich mit verschiedenen Einrichtungen zusammenzutun, um über eine Einzelförderung dieser Entwicklung in einem Drittmittelprojekt hinaus dauerhaft die betreffende Forschungssoftware pflegen und bereitstellen zu können. Die Kosten für einen größeren Entwicklungssprung (oder die Erstentwicklung) werden über einen gemeinsamen Drittmittelantrag realisiert, die Kosten für die dauerhafte Bereitstellung und Pflege teilen sich dann die den Antrag stellenden Institutionen.

Die von uns umrissenen Modelle sind nur eine Auswahl verschiedener möglicher, die uns gleichwohl für die in der Kommunikations- und Medienwissenschaft aktuell bestehende Forschungssoftware sinnvoll erscheinen. Insgesamt stehen diese Modelle für einen allgemeineren Zusammenhang, nämlich dass die Finanzierung von Forschungssoftware als eine **kollektive Aufgabe des Fachs** zu begreifen ist.

Anerkennung der Entwicklung von Forschungssoftware

Auch wenn so entsprechende finanzielle Ressourcen für die Entwicklung von Forschungssoftware gesichert wären, wird deren Entwicklung doch eine spezielle Aufgabe einzelner Expert*innen bleiben. Deren Leistung an der Schnittstelle von empirischer Forschung und Softwareentwicklung gilt es entsprechend als einen **wichtigen Beitrag für eine sich verändernde Kommunikations- und Medienwissenschaft** anzuerkennen. Wir empfehlen daher, dass auch Forschungssoftware in Veröffentlichungen zitiert werden sollte, inklusive der Namensnennung ihrer Entwickler*innen. Dies sollte von Zeitschriften, Herausgeber*innen und Reviewer*innen eingefordert werden. Entwickler*innen sollten umgekehrt entsprechende Zitationsinformationen in Repositorien zur Verfügung stellen, damit die Nutzer*innen auf einfache Weise auf die Urheber*innen der Forschungssoftware hinweisen können.

Forschungssoftware sollte nach der Veröffentlichung auch durch Publikationen **in Fachzeitschriften sichtbar** gemacht werden. Insbesondere Fachzeitschriften mit Methoden-Schwerpunkten (z.B. *Communication Methods and Measures* oder *Computational Communication Research*) bieten bereits die Möglichkeit, Software zu publizieren. Darüber hinaus sollte über weitere Publikationsformate nachgedacht werden. Denkbar wären beispielsweise Softwarevorstellungen in den DGPuK-Publikationsorganen seitens der Entwickler*innen oder die bereits genannten Softwarerezensionen durch die Fachcommunity (analog zu den Buchbesprechungen in vielen Zeitschriften).

Schließlich möchten wir **Auszeichnungen und Preise** für Forschungssoftware-Entwicklung anregen, um Anerkennung zu erhöhen und mehr Bewusstsein im Fach selbst zu schaffen, z.B. im Rahmen der Jahrestagung der DGPuK. Die entsprechende Bildung von Preis-Komitees würde zusätzlich nicht nur sicherstellen, dass eine fortlaufende Sammlung neuer Projekte stattfindet, sondern auch, dass die Institutionalisierung der Software-Entwicklung vorangetrieben wird.

Damit sehen wir bei verschiedenen Bezugsgruppen unterschiedliche Anforderungen im Hinblick auf die Finanzierung von Forschungssoftware und die Anerkennung ihrer Entwickler*innen:

- **Entwickler*innen und Forscher*innen** sollten bei der Finanzierung von Forschungssoftware kooperative Wege gehen und Modelle vorantreiben, die kollektiv eine dauerhafte Absicherung der Entwicklung von Forschungssoftware sicherstellen.
- Die **Wissenschaftsförderung** sollte spezielle Programme zur Förderung fachspezifischer Forschungssoftware auflegen und auch in den Normalverfahren die Entwicklung nachhaltiger Finanzierungsmodelle im Blick haben.
- In der **Fachgesellschaft** sollte ein Austausch über kooperative Finanzierungsmodelle erfolgen und die Entwicklung von Forschungssoftware sollte besser und sichtbarer anerkannt werden.
- **Fachzeitschriften** sollten strikt darauf achten, dass bei der Arbeit mit bestimmter Forschungssoftware auch deren Entwickler*innen angemessen zitiert werden.
- In der **Lehre und Qualifikation** gilt es, ein Verständnis für die Kosten und Leistung der Entwicklung von Forschungssoftware zu vermitteln.

5. Kompetenz und Vermittlung

Die bisherigen Empfehlungen gehen von ausreichender Kenntnis digitaler Methoden und der Computational Social Sciences in der Kommunikations- und Medienwissenschaft aus. Um diese Kompetenzen einerseits sicherzustellen, andererseits auch mittel- und langfristig zu fördern, ist ein Umdenken in Bezug auf Lehre und Förderung von Jungwissenschaftler*innen nötig. Dabei sollen Wissenschaftler*innen in Qualifikationsphasen nicht nur in die Lage versetzt werden, Forschungssoftware zu nutzen, sondern es sollen auch Anreize geschaffen werden, Forschungssoftware anzupassen und zu entwickeln.

Abbau von Hürden

Dem Vorhaben, Forschungssoftware in die kommunikations- und medienwissenschaftliche Lehre und Nachwuchsausbildung zu integrieren, stehen verschiedene Hindernisse entgegen. Oft erschweren die Beharrungskraft bisheriger Methodensozialisation und die individuelle sowie institutionelle Trägheit die notwendigen Änderungen der Prüfungsordnungen und Modulkataloge in den Studiengängen bzw. die Möglichkeiten der Promotionsausbildung. Aufgrund von fehlendem Wissen besteht immer wieder eine Skepsis gegenüber den Erkenntnismöglichkeiten beim Einsatz von Forschungssoftware – oder umgekehrt werden die Möglichkeiten des Einsatzes von Forschungssoftware überschätzt. Die im Einzelfall in Bezug auf bestimmte Fragestellungen bestehende Komplexität bei der Operationalisierung von Forschungssoftware sowie die teils fehlende Passung von Forschungssoftware auf kommunikations- und medienwissenschaftliche Forschungsdesiderate können weitere Hürden darstellen.

Um diese Schwellen abzubauen, sollte sich die Kommunikations- und Medienwissenschaft den Herausforderungen stellen, die Kooperation, das Teilen und das wechselseitige Vermitteln des Einsatzes von Forschungssoftware in die alltägliche Praxis zu integrieren. Interdisziplinäre Kooperationen und kollaborative Szenarien für die Ausbildung (z.B. Co-Teaching und Peer-Learning) können helfen, einen Einstieg und erste Orientierung in die Vermittlung dieser Kompetenzen zu bieten. Um mit dem rapiden Wandel in Software und Technik mitzuhalten, müssen kontinuierliche Vermittlungsstrukturen geschaffen werden (vgl. Koenen, 2021).

Box 7: Arbeitsfelder im Bereich der Computational Social Science in der Lehre

Im Zusammenspiel mit Fach- und Methodenkompetenzen entfalten die aufgezeigten Forschungssoftware-Kompetenzen eine besondere Relevanz im Bereich Computational Communication Science (CCS) (Domahidi, Yang, Niemann-Lenz, & Reinecke, 2019). Für die Lehre im Bereich CCS ist die Vermittlung von diesem erweiterten „Kompetenzpool“ zentral, da er nicht nur im akademischen Bereich, sondern auch in der Praxis ausgesprochen gefragt ist (Stockinger, Stadelmann & Ruckstuhl, 2016). Um Verfahren und Techniken der CCS bereits in der Lehre zu vermitteln (Lazer et al., 2009; van Atteveldt & Peng, 2018) müssen bestehende Lehrpläne erweitert werden. Ein exemplarisches Beispiel zur Vermittlung von Anwendungs- und Kritikkompetenz liegt im Bereich automatisierter Datenerfassung über Webscraping und APIs: Im Rahmen der CCS Lehre sollten in Verbindung mit relevanten Forschungsfragen und methodischen Herangehensweisen unterschiedliche Softwarelösungen ausprobiert werden (z.B. Facepager, flexibles Scraping oder API-Direktzugriff) und ihre Vor- und Nachteile, Implikationen für den gesamten Forschungsprozess sowie die Validität der Daten diskutiert werden. Die AG „CCS in der Lehre“ widmet sich folgerichtig der Frage, wie eine solche Lehre im Spannungsfeld zwischen Fach-, Methoden- und Forschungssoftware-Kompetenzen gestaltet werden kann.

Notwendige Kompetenzbereiche

Inhalte, die Studierenden und Promovierenden einen angemessenen Umgang mit Forschungssoftware vermitteln, müssen in die Curricula der kommunikations- und medienwissenschaftlichen Studiengänge Einzug halten. Dabei stellt sich für Lehrverantwortliche die Frage, welche Inhalte in welchem curricularen Zusammenhang mit welchem Ziel vermittelt werden sollen: Eine nachhaltige Qualifikation braucht konkrete Vorstellungen vom Umfang digitaler Kompetenzen bezüglich Forschungssoftware, wobei diese notwendigen Kompetenzen über bloße Anwendungskenntnisse hinausgehen. Der Umgang mit Forschungssoftware lässt sich insbesondere auf vier Kompetenzbereiche aufteilen, die aus unserer Sicht auch in dieser Priorisierung aufeinander aufbauen:

Im Kern der **Softwarekompetenz** steht ein Grundwissen über die Spezifika digitaler Technologien und Prozesse, also über den Kontext, in den Forschungssoftware eingebettet ist. Notwendig für den Einsatz und die Entwicklung von Forschungssoftware ist ein gutes Verständnis der Grenzen und Möglichkeiten von Computersoftware. Dazu gehört insbesondere Grundwissen über die Architektur von Betriebssystemen, über Rechnernetze und über zentrale Aspekte der IT-Sicherheit. So sollen Unterschiede deutlich werden, die sich bereits aus der Wahl einer Programmiersprache oder der Genese bestimmter Daten ergeben. Der Anspruch der Ausbildung soll dabei keineswegs Informatik-Charakter bekommen, sondern vielmehr die Grundlagen dafür schaffen, informatische Probleme abstrahieren zu können, um sie so handhabbar zu machen (vgl. Bradshaw, 2018).

Bei der **Anwendungskompetenz** steht die praktische Anwendung von bestehender Forschungssoftware im Fokus. Zentral soll hierbei sein, dass die Nutzung von Forschungssoftware nicht beiläufig, sondern bewusst geschehen muss. Dazu zählt der Umgang mit Dokumentationen sowie die transparente Nutzung von Forschungssoftware, also das Zitieren und Dokumentieren relevanter Informationen über die genutzte Software. Zu diesen relevanten Informationen zählen beispielsweise die Version der Software, die tatsächlich genutzten Funktionen und spezifische Konfigurationen der Software.

Neben der spezifischen Anwendungs- und der allgemeinen Softwarekompetenz bedarf es ferner einer **Kritikkompetenz**, also der Fähigkeit, Kritik an bestehender Forschungssoftware zu üben (vgl. Khoo et al., 2017; van Es, Schäfer & Wieringa 2021). Dabei steht im Vordergrund, die Eignung einer Forschungssoftware für eine bestimmte Aufgabe zu bewerten und diese Einschätzung auf eine Reihe von verschiedenen Softwarelösungen anzuwenden, die für einen ähnlichen Zweck entwickelt wurden. Eine solche Kompetenz ist einerseits für wissenschaftliche Arbeiten generell nötig, um die Erkenntnisleistungen verschiedener softwaregestützter Methoden einzuschätzen und einzuordnen; andererseits versetzt sie auch in die Lage, Forschungssoftware im Rahmen von Fachzeitschriften begutachten und besprechen zu können.

Zuletzt sehen wir den Bedarf einer **Entwicklungskompetenz**, worunter die grundsätzliche Fähigkeit des Programmierens, aber auch das Verständnis unterschiedlicher Programmierparadigma (z.B. prozedurale, funktionale und objektorientierte Programmierung) zu fassen ist. Dazu zählen zunächst ein Verständnis von Variablen, Typen, (verzweigten) Abfragen und Schleifen, in weiterer Folge die Fähigkeit Programme zu modularisieren, der Umgang mit Programmbibliotheken und die Einrichtung von kollaborativen Entwicklungsumgebungen. Zentrale Bestandteile der Entwicklungskompetenz sind darüber hinaus ein nachvollziehbarer und lesbarer Programmierstil, angemessene Dokumentation und Tests zentraler Funktionen. Diese Entwicklungskompetenz wird nicht nur dann relevant, wenn Forscher*innen selbst komplexe Forschungssoftware entwickeln, sondern auch dann, wenn sie bestehende Software an ihre eigenen Bedürfnisse anpassen wollen (vgl. Haim, 2021).

Insgesamt erscheinen uns vor diesem Hintergrund folgende Punkte wichtig:

- Für **Entwickler*innen** gilt, dass sie bei der Programmierung von Forschungssoftware auch die Lehre im Blick haben sollten und so irgend möglich neben der reinen Dokumentation des Codes auch an kurze Anleitungen zur jeweiligen Forschungssoftware denken sollten.
- **Anwender*innen** von Forschungssoftware sollten ihre Erfahrungen im Umgang mit spezifischer Software anderen zur Verfügung stellen, etwa über Erfahrungsberichte und Tutorials, und sich an der Qualitätssicherung der Software beteiligen, etwa über Gutachten in Fachzeitschriften (z.B. im Journal of Open-Source-Software).
- Die **Fachgesellschaft** sollte den fortlaufenden Diskurs um den Einsatz von Forschungssoftware in der Lehre und Nachwuchsausbildung fördern.
- Im Bereich des **Wissenschaftsmanagements** ist es zentral, dass neben digitalen Methoden bzw. Computational Social Sciences auch die notwendigen Kompetenzen im Hinblick auf Forschungssoftware in die (Weiter-)Entwicklung von Curricula Eingang finden.
- Generell sollten alle Kolleg*innen, die in der **Lehre und Qualifikation** aktiv sind, das Thema der Forschungssoftware zu einem festen Bestandteil der Studiengänge und Nachwuchsförderung machen.

6. Fazit: Für eine nachhaltige Entwicklung von Forschungssoftware

Ziel dieses Papiers war es erstens, für die Kommunikations- und Medienwissenschaft den Stand, die Herausforderungen und Perspektiven zu beschreiben, die die Verankerung von Forschungssoftware für das Fach bedeuten. Zweitens sollten daraus Empfehlungen für Wissenschaftler*innen, Entwickler*innen, Mittelgeber*innen, das Wissenschaftsmanagement und die Fachgesellschaft abgeleitet werden. Im Kern lassen sich die von uns formulierten Punkte dahingehend zusammenfassen, dass es insbesondere um eine **nachhaltige Entwicklung und Nutzung von Forschungssoftware im Fach** geht. Im Vordergrund sollte solche Forschungssoftware stehen, die die Forschungsfragen der Kommunikations- und Medienwissenschaft zu bearbeiten hilft. Wichtig erscheint uns darüber hinaus, einerseits die Kreativität und Ergebnisoffenheit der Forschung auch mittels Forschungssoftware zu wahren, andererseits gerade wegen der in Teilen umfassenden Ressourcen, die in deren Entwicklung fließt, Insellösungen zu vermeiden. Nachhaltig wird die Entwicklung von Forschungssoftware für die Fachgemeinschaft dann, wenn sich um einzelne „Software-Lösungen“ Gruppen von Forscher*innen bilden, die diese dauerhaft absichern, dabei Dokumentation und Qualität im Blick haben, wie auch die sich verändernden Forschungsfragen und Gegenstände des Fachs.

Versteht man dies als gemeinsames Ziel, ist für die nachhaltige Entwicklung von Forschungssoftware vielleicht mehr noch als in anderen Bereichen der Forschung eine **umfassende Kooperation** die zentrale Basis. Stichworte, die wir in diesem Papier genannt haben, sind dabei Open-Source bei der Entwicklung, die Unterstützung offener Formate, kooperative Modelle der Softwareentwicklung (mit einer umfassenden Zusammenarbeit auch mit anderen Disziplinen), das Zusammenarbeiten in und mit Gruppen von Nutzer*innen, das Teilen von Kosten sowie gemeinsames Lehren bei der Verankerung von Forschungssoftware in der Lehre und Qualifikation. Unser Ziel ist es, mit diesem Papier in der DGPuK einen Anstoß zu einer solchen Kooperation zu geben.

Literatur

Anzt, H., Bach, F., Druskat, S., Löffler, F., Loewe, A., Renard, B. Y., . . . Weeber, R. (2020). An environment for sustainable research software in Germany and beyond: current state, open challenges, and call for action. *F1000Research*, *9*, 295. doi:10.12688/f1000research.23224.1

Bradshaw, P. (2018). Data journalism teaching, fast and slow. Asia Pacific Media Educator, 28(1), 55-66. https://journals.sagepub.com/doi/full/10.1177/1326365X18769395.

Brosius, H.-B. (2016). Warum Kommunikation im Internet öffentlich ist. *Publizistik*, *61*(4), 363-372. doi:10.1007/s11616-016-0304-6

Chadwick, A. (2017). *The hybrid media system: Politics and power*. Second edition. Oxford: Oxford University Press.

Couldry, N., & Mejias, U. (2019). *The costs of connection. How data is colonizing human life and appropriating it for capitalism*. Stanford, CA: Stanford UP.

Diakopoulos, N. (2019). *Automating the News*. Cambridge: Harvard University Press.

Domahidi, E., Yang, J.W., Niemann-Lenz, J., & Reinecke, L. (2019). Computational Communication Science | Outlining the Way Ahead in Computational Communication Science: An Introduction to the IJoC Special Section on "Computational Methods for Communication Science: Toward a Strategic Roadmap". International Journal of Communication 13 (2019), 3876-3884

Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Daumé III, H., & Crawford, K. (2020). Datasheets for Datasets. ArXiv:1803.09010 [Cs]. http://arxiv.org/abs/1803.09010.

Haim, M. (2021). Gütekriterien und Handlungsempfehlungen für die Entwicklung von Forschungssoftware in der Kommunikations- und Medienwissenschaft. Medien & Kommunikationswissenschaft, 69(1), 65-79. doi:10.5771/1615-634X-2021-1-65

Hasebrink, U., & Hepp, A. (2017). How to research cross-media practices? Investigating media repertoires and media ensembles. *Convergence*, *23*(4), 362-377.

Hasselbring, W., Carr, L, Hettrick, S., Packer, H. & Tiropanis, T. (2020). From FAIR research data toward FAIR and open software. In: *it – Information Technology*, 62(1), 39-47. https://doi.org/10.1515/itit-2019-0040.

Hepp, A. (2016). Kommunikations- und Medienwissenschaft in datengetriebenen Zeiten. *Publizistik*, *61*(3), 225-246.

Hepp, Andreas; Loosen, Wiebke; Hasebrink, Uwe (2021). Jenseits des Computational Turn: Methodenentwicklung und Forschungssoftware in der Kommunikations- und Medienwissenschaft. In: *Medien & Kommunikationswissenschaft*, 69(1), S. 4-24.

Hogan, B. (2018). Social media giveth, social media taketh away: Facebook, friendships and APIs. *International Journal of Communication*, *12*, 592-611.

Hohmann, F. (2021). Co-Creation als Entwicklungsmethode: Zu Möglichkeiten und Grenzen partizipativer Forschungssoftwareentwicklung am Beispiel der Sortiersoftware MeSort und Tagebuchsoftware MeTag. *Medien & Kommunikationswissenschaft*, *69*(1), 97-116.

Jarren, O. (2016). Nicht Daten, sondern Institutionen fordern die Publizistik- und Kommunikationswissenschaft heraus. *Publizistik*, *61*(4), 373-383. doi:10.1007/s11616-016-0301-9

Johanson, A., & Hasselbring, W. (2018). Software engineering for computational science: Past, present, future. *Computing in Science & Engineering*. Retrieved from https://ieeexplore.ieee.org/abstract/document/8254315.

Katerbow, M., & Feulner, G. (2018). *Handreichung zum Umgang mit Forschungssoftware*. Bonn, Potsdam: DFG, PIK. Retrieved from http://doi.org/10.5281/zenodo.1172970.

Khoo, E., Hight, C., Torrens, R., & Cowie, B. (2017). Introduction: Software and other literacies. In E. Khoo, C. Hight, R. Torrens, & B. Cowie (Eds.), *Software Literacy* (pp. 1-14). Singapore: Springer. Retrieved from https://doi.org/10.1007/978-981-10-7059-4_1.

Koenen, E. (2021). Forschungssoftware für die Kommunikations- und Mediengeschichte. Epistemologische Herausforderungen und Perspektiven. *Medien & Kommunikationswissenschaft*, *69*(1), 117-135.

Krüger, U., & Meyen, M. (2018). Auf dem Weg in die Postwachstumsgesellschaft. Plädoyer für eine transformative Kommunikationswissenschaft. *Publizistik*, *63*(3), 341-357. doi:10.1007/s11616-018-0424-2

Lazer, D., Pentland, A., Adamic, L., Aral, S., Barabasi, A.-L., Brewer, D., . . . Gutmann, M. (2009). Life in the network: The coming age of computational social science. *Science*, *323*(5915), 721-723. Retrieved from https://europepmc.org/articles/pmc2745217.

Manovich, L. (2020). *Cultural analytics*. Cambridge MA, London: MIT Press.

Martin, Robert C. (2008). Clean Code: A Handbook of Agile Software Craftsmanship, Pearson.

Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D., & Gebru, T. (2019). Model Cards for Model Reporting. Proceedings of the Conference on Fairness, Accountability, and Transparency, 220–229. https://doi.org/10.1145/3287560.3287596

Puschmann, C., & Ausserhofer, J. (2017). Social data APIs: Origin, types, issues. In M. T. Schäfer & K. van Es (Eds.), *The Datafied Society. Studying Culture through Data* (pp. 147-154). Amsterdam: Amsterdam University Press.

Rogers, R. (2021). *Digitale Methoden: Zur Positionierung eines Ansatzes*. In: *Medien & Kommunikationswissenschaft*, 69(1), S. 1-21

Sanders, E. B.-N., & Stappers, P. J. (2008). Co-creation and the new landscapes of design. *CoDesign*, *4*(1), 5-18. doi:10.1080/15710880701875068

Sharp, H., Preece, J., & Rogers, Y. (2019). *Interaction Design*. Cambridge: John Wiley & Sons. Retrieved from https://play.google.com/store/books/details?id=AlSQDwAAQBAJ&source=gbs_api.

Simonsen, J., & Robertson, T. (Eds.). (2013). *Routledge international handbook of participatory design*. New York: Routledge.

Stockinger, K., Stadelmann, T. & Ruckstuhl, A. (2016). Data Scientist als Beruf. In: Fasel, D. et al. (Hrsg.). *Big Data. Grundlagen, Systeme und Nutzungspotenziale*, Wiesbaden, S. 59-81.

Storer, T. (2017). Bridging the Chasm. *ACM Computing Surveys*, *50*(4), 1-32. doi:10.1145/3084225

Strippel, C., Bock, A., Katzenbach, C., Mahrt, M., Merten, L., Nuernbergk, C., . . . Waldherr, A. (2018). Die Zukunft der Kommunikationswissenschaft ist schon da, sie ist nur ungleich verteilt. Eine Kollektivreplik. *Publizistik*, 63, 11-27.

Strippel, C. (2021): Forschungsinfrastrukturen für die Kommunikations- und Medienforschung im deutschsprachigen Raum. Initiativen, Bedarfe und Perspektiven. Medien & Kommunikationswissenschaft, 69(1), 136-157.

Struck, A., Loewe, A., Achhammer, E., Rack, F., Bach, F., Löffler, F., Seemann, G., Anzt, H., Funk, M., Unger, S., Druskat, S. & Friedl, S. (2020). *A Guide for Publishing, Using, and Licensing Research Software in Germany*. Online unter https://doi.org/10.5281/zenodo.4327147 [02.02.2021].

Theis-Berglmair, A. M. (2016). Auf dem Weg zu einer Kommunikationswissenschaft. *Publizistik*, *61*(4), 385-391. doi:10.1007/s11616-016-0302-8

van Atteveldt, W., & Peng, T.-Q. (2018). When Communication Meets Computation: Opportunities, Challenges, and Pitfalls in Computational Communication Science. *Communication Methods and Measures*, *12*(2–3), 81–92. https://doi.org/10.1080/19312458.2018.1458084.

van Es, K., Schäfer, M. T. & Wieringa, M. (2021). Tool Criticism and the Computational Turn. A "Methodological Moment" in Media and Communication Studies. *Medien & Kommunikationswissenschaft*, 69(1), 46-64.

Veltri, G. A. (2020). *Digital social research*. Cambridge: Polity.

Vendome, C., German, D. M., Di Penta, M., Bavota, G., Linares-Vásquez, M., & Poshyvanyk, D. (2018). *To distribute or not to distribute*. Proceedings from Proceedings of the 40th International Conference on Software Engineering, New York, NY, USA.

von Nordheim, G., Koppers, L., Boczek, K., Rieger, J., Jentsch, C., Müller, H. Rahnenführer, J. (2021). Die Entwicklung von Forschungssoftware als praktische Interdisziplinarität. Medien und Kommunikationswissenschaft(M&K),1/2021.

Ziemann, M., Eren, Y., & El-Osta, A. (2016). Gene name errors are widespread in the scientific literature. *Genome Biol*, *17*(1), 177. doi:10.1186/s13059-016-1044-7

Zuboff, S. (2018). *Das Zeitalter des Überwachungskapitalismus*. Frankfurt a.M., New York: Campus.