HELMHOLTZ GEMEINSCHAFT

Open Science

Helmholtz Open Science Workshop „Zugang zu und Nachnutzung von wissenschaftlicher Software“ #hgfos16

Report

November 2016

Impressum

Die Onlineversion dieser Publikation finden Sie unter:
http://doi.org/10.2312/lis.17.01

Autorinnen und Autoren
Kaja Scheliga, Heinz Pampel, Erik Bernstein, Christoph Bruch, Wolfgang zu Castell, Markus Diesmann, Bernadette Fritzsch, Jürgen Fuhrmann, Holger Haas, Martin Hammitzsch, David Lähnemann, Alice McHardy, Uwe Konrad, Gianna Scharnberg, Andreas Schreiber, Dirk Steglich

Redaktion
Kaja Scheliga (Helmholtz-Gemeinschaft)
Heinz Pampel (Helmholtz-Gemeinschaft)

Kontakt
Helmholtz Open Science Koordinationsbüro
c/o Helmholtz-Zentrum Potsdam
Deutsches GeoForschungsZentrum GFZ
Telegrafenberg, 14471 Potsdam
E-Mail: open-science@helmholtz.de

Stand
März 2017

Inhaltsverzeichnis

1. Zusammenfassung

Der Report des Helmholtz Open Science Workshops „Zugang zu und Nachnutzung von wissenschaftlicher Software“ #hgfos16 behandelt die Themen Standards und Qualitätssicherung; Reproduzierbarkeit; Lizenzierung und weitere rechtliche Aspekte; Zitation und Anerkennung; Sichtbarkeit und Modularität; Geschäftsmodelle; Personal, Ausbildung, Karrierewege. Diese Themen sind eng miteinander verzahnt. Für jeden Themenbereich werden jeweils die Relevanz, Fragestellungen, Herausforderungen, mögliche Lösungsansätze und Handlungsempfehlungen betrachtet.

Technische Infrastrukturen sind das Fundament für den Zugang zu und die Nachnutzung von wissenschaftlicher Software sowie für ihre Nachhaltigkeit. Die Balance zwischen der Nutzung eigener und kommerzieller Infrastrukturen ist eine Herausforderung bei wissenschaftlicher Softwareentwicklung. Die Bereitstellung eigener Infrastrukturen kann die Autonomie der wissenschaftlichen Einrichtungen gewährleisten. Gleichzeitig können eigene Infrastrukturen durch die Nutzung von Diensten kommerzieller Anbieter ergänzt werden und so die Verfügbarmachung von wissenschaftlicher Software unterstützen. Die langfristige Finanzierung und Förderung technischer Infrastrukturen ist von großer Bedeutung.

Standards und Qualitätssicherung sind elementar für die Zugänglichmachung und Nachnutzung von Software. Die Anwendung etablierter Standards fördert die Nachvollziehbarkeit und die Interoperabilität des entwickelten Softwarecodes. In der Praxis sind allgemeine Mindestanforderungen allerdings schwer so zu definieren, dass sie den diversen disziplinären Anwendungen gerecht werden. Bei der Qualitätskontrolle gibt es noch viele manuelle Schritte, die zeitaufwendig und fehleranfällig sind. Style-Guides für verschiedene Programmiersprachen, Automatisierungsmechanismen und der Wissenstransfer zwischen verschiedenen Communities sowie Code-Review-Verfahren sind Ansätze, um Standards zu etablieren und die Qualitätssicherung zu fördern.

Reproduzierbarkeit ist durch das Zusammenspiel von Software und entsprechender technischer Infrastruktur in Verbindung mit Forschungsdaten und Workflows gekennzeichnet. Es gilt den notwendigen und den gewünschten Zeitraum der Reproduzierbarkeit von Software sowie die technische Realisierbarkeit und die verschiedenen Ebenen der Reproduzierbarkeit abzuwägen. Versionskontrolle von Software, ausführliche Dokumentation und die Veröffentlichung von Software unter Verwendung von standardisierten und offenen Lizenzen können die Reproduzierbarkeit fördern.

Lizenzierung und weitere rechtliche Aspekte umfassen das Spannungsfeld in Bezug auf die Verwertungsrechte von Software. Die kommerzielle Verwertung von Software ist für einige Einrichtungen von Interesse. Die Nutzung offener Lizenzen kann den Austausch von Ideen befördern und eine rechtssichere Grundlage für Zusammenarbeit bilden. Beratungsangebote (z. B. Informationsbroschüren, Checklisten, Dokumentationen von Kriterien, Best-Practices sowie Beratung zu Lizenzfragen durch Spezialist/innen an wissenschaftlichen Einrichtungen oder in Anwaltskanzleien) können dazu beitragen, wissenschaftliche Software im Sinne der Zugänglichkeit und Nachnutzung rechtssicher zu veröffentlichen.

Zitation und Anerkennung sind wichtige Faktoren bei der Veröffentlichung von Software. In der Wissenschaft wird die Leistung der Software-Entwickler/innen nicht genügend anerkannt und es fehlt an etablierten Publikations- und Zitationsstrategien für wissenschaftliche Software. Wissenschaftliche Software sollte ein eigener Publikationstyp sein, der als eigenständiges, eindeutig identifizierbares (z. B. durch die Verwendung des Digital Object Identifier - DOI) wissenschaftliches Produkt wahrgenommen, geschätzt und anerkannt wird.

Sichtbarkeit und Modularität erleichtern die nachhaltige Nutzung der Software und die Nachnutzung des Codes. Modularität erfordert einen zusätzlichen Aufwand bei der Softwareentwicklung, der auch bei der Planung und Finanzierung berücksichtigt werden sollte. Die Verwendung von Standards kann zur Sichtbarkeit beitragen; eine ausführliche Dokumentation der Software ist förderlich für die Nachnutzung/Weiterentwicklung.

Geschäftsmodelle für wissenschaftliche Software können zur Instandhaltung und Weiterentwicklung über die Projektförderung hinaus beitragen. Verwertungsstrategien können sowohl dem Zugang zu und der Nachnutzung von wissenschaftlicher Software dienen als auch einen Beitrag zur finanziellen Absicherung der Software leisten. Das potenzielle Spannungsfeld zwischen kommerzieller Verwertung und der geforderten Offenheit im Sinne von Open Science muss bei Verwertungsstrategien bedacht werden.

Personal, Ausbildung, Karrierewege sind zentral für die Zukunft von wissenschaftlicher Software. Aktuell mangelt es an Aus- und Weiterbildungsformaten, die Nachwuchswissenschaftler/innen die entsprechenden Kenntnisse für Softwareentwicklung in der Wissenschaft vermitteln. Es fehlt an Karriereperspektiven und Wertschätzung für Softwareentwicklung im Wissenschaftsbetrieb. Um Softwareentwicklung zu fördern, sind vielfältige Lehr- und Lernformate sowie diverse Aus- und Weiterbildungsformen nötig.

In **Keynotes und Impulsvorträgen** wurden auf dem Workshop verschiedene Perspektiven rund um den Zugang zu und die Nachnutzung von wissenschaftlicher Software thematisiert (vgl. Abstracts im Anhang). In seiner Keynote hat Dr. Johannes Köster (Centrum Wiskunde & Informatica Amsterdam) Wege aus der in-silico-Reproduzierbarkeitskrise am Beispiel der Bioinformatik diskutiert. Dr. Sünje Dallmeier-Tiessen (CERN) hat in ihrer Keynote über Reproduzierbarkeit und Open Science mit besonderem Augenmerk auf Software referiert und dabei Erfahrungsberichte aus dem Bereich der Hochenergiephysik vorgestellt. In Impulsvorträgen wurde die Nachhaltigkeit von Forschungssoftware aus Sicht der Deutschen Forschungsgemeinschaft (DFG) präsentiert, ein Entwurf eines Metadatenrepositoriums zur Erfassung technischer Nachhaltigkeit von Forschungssoftware diskutiert, Qualitätsmanagement von und Infrastruktur für Open-Source-Software in der Wissenschaft am Beispiel des Medical Imaging Interaction Toolkits (MITK) vorgestellt, und an Hand der Bioinformatik-Software BALL wurde erörtert, wie die Pflege von wissenschaftlicher Software aussehen kann.

2. Einleitung

Mit der voranschreitenden Digitalisierung von Forschung und Lehre steigt die Zahl an Software-Lösungen, die an wissenschaftlichen Einrichtungen entstehen und zur Verarbeitung wissenschaftlicher Daten genutzt werden. Die unter dem Stichwort Open Science geforderte Zugänglichkeit und Nachnutzung von wissenschaftlichen Ergebnissen kann in vielen Fachgebieten nur sichergestellt werden, wenn neben den Forschungsdaten auch der Programmcode offen zugänglich gemacht wird.

Die Task Group „Zugang zu und Nachnutzung von wissenschaftlicher Software“ des Arbeitskreises Open Science der Helmholtz-Gemeinschaft veranstaltete vom 22.-23. November 2016 am Helmholtz-Zentrum Dresden-Rossendorf einen Workshop, der sich mit folgenden Aspekten rund um die Zugänglichmachung und Nachnutzung von wissenschaftlicher Software befasste:

- Technische Infrastrukturen
- Standards und Qualitätssicherung
- Reproduzierbarkeit
- Lizenzierung und weitere rechtliche Aspekte
- Zitation und Anerkennung
- Sichtbarkeit und Modularität
- Geschäftsmodelle
- Personal, Ausbildung, Karrierewege

An dem Workshop nahmen 70 Personen aus Wissenschaft, Infrastruktur und Administration wissenschaftlicher Einrichtungen in Deutschland teil.[1] Der Workshop wurde vom Helmholtz Open Science Koordinationsbüro und dem Helmholtz-Zentrum Dresden-Rossendorf organisiert.

In den folgenden Abschnitten des Reports werden Relevanz, Fragestellungen, Herausforderungen, Lösungsansätze und Handlungsempfehlungen der in den Sessions des Workshops diskutierten Themen an Hand der jeweiligen Protokolle zusammengefasst. Die Fragestellungen in den jeweiligen Themenbereichen sind als Denkimpulse zu verstehen. Die Themenbereiche sind eng miteinander verzahnt, gleichzeitig können die einzelnen Themenabschnitte unabhängig voneinander gelesen werden.

Der vorliegende Report erhebt keinen Anspruch auf die vollständige Wiedergabe der auf dem Workshop diskutierten Aspekte.

[1] Siehe Liste der Teilnehmer/innen im Abschnitt 12.3.

3. Technische Infrastrukturen

Relevanz

Technische Infrastrukturen sind das Fundament für den Zugang zu und die Nachnutzung von wissenschaftlicher Software sowie für ihre Nachhaltigkeit. Sie unterstützen die Entwickler/innen im kollaborativen Prozess der Spezifikation, des Entwurfs, des Tests und der Dokumentation von Software und stellen Repositorien für die langfristige Archivierung und Bereitstellung der verschiedenen Software-Versionen bereit.

Fragestellungen

- Welche Infrastrukturen werden für nachhaltige Softwareentwicklung und -pflege benötigt?
- Können eigene Software-Infrastrukturen aufgebaut werden und wie nachhaltig sind diese?
- Welche Rolle spielt die Cloud für Versionskontrolle, Dokumentation und Archivierung?
- Welche Automatisierungs- und Testwerkzeuge gibt es?
- Wie kann die Finanzierung von technischen Infrastrukturen sichergestellt werden?

Herausforderungen

Technische Infrastrukturen wie Plattformen zur Softwareentwicklung oder Repositorien zur dauerhaften Speicherung des Programmcodes müssen nachhaltig finanziert werden und von qualifiziertem Personal betreut werden. In der Praxis fehlt es häufig an den entsprechenden Mitteln. Die Nutzung von Cloud-Diensten wie z. B. GitHub erleichtert die institutionsübergreifende und internationale Zusammenarbeit bei der Entwicklung und Nachnutzung von wissenschaftlicher Software. Gleichzeitig birgt die Nutzung und der Verlass auf externe Infrastrukturdienstleister zur kollaborativen Softwareentwicklung die Gefahr der Entstehung von Abhängigkeiten.
In Verbindung mit technischen Infrastrukturen sind auch Automatisierungs- und Testwerkzeuge ein wichtiges Thema, um die Zusammenarbeit verteilter Teams zu verbessern und die Qualität und Nachvollziehbarkeit der Lösungen sicherzustellen.

Lösungsansätze

Die Bereitstellung und Pflege von eigenen technischen Infrastrukturen durch wissenschaftliche Einrichtungen ist ein Lösungsansatz, der die Autonomie der wissenschaftlichen Einrichtungen gewährleisten kann, gleichzeitig aber aufwendig ist. Für große Entwicklungsprojekte ist es notwendig, lokale Infrastrukturen für die Entwicklung und den Test vorzuhalten und diese mit den spezifischen Datenrepositorien und Analysetools zu vernetzen. Das sichert nicht nur die Qualität und Nachvollziehbarkeit der Software, sondern ermöglicht neuen Entwickler/innen im Team einen schnelleren Einstieg. Eigene Infrastrukturen können mit Cloud-Diensten externer Infrastrukturdienstleister (zum Beispiel GitHub oder GitLab) vernetzt und so einem breiteren Anwender/innenkreis zugänglich gemacht werden. Cloud-Infrastrukturen für Software haben Grenzen in Bezug auf Ablauf- und Testumgebungen. Für viele Projekte kann die Bereitstellung von Containern für diese Umgebungen helfen, die Nachhaltigkeit der Projekte zu verbessern. Eine solide, vertrauenswürdige, sichere und nutzerfreundlich gestaltete technische Infrastruktur kann ein Anreiz für die Verfügbarmachung von Programmcode an einer wissenschaftlichen Einrichtung sein.

Beispiele für Software-Infrastrukturen:

- CERN Analysis Preservation[2]
- CERNVM[3]
- Docker[4]
- GitLab-Dienst an der Technischen Universität Berlin[5]
- INSPIRE[6]
- Kopplung von Zenodo mit GitHub im Rahmen von OpenAIRE[7]
- Software Engineering Initiative beim Deutschen Zentrum für Luft- und Raumfahrt
- Software Sustainability Institute[8]
- Travis CI[9]

[2] https://analysis-preservation.cern.ch
[3] https://cernvm.cern.ch
[4] https://www.docker.com/
[5] https://www.tubit.tu-berlin.de/menue/dienste/daten_server/gitlab-dienst/
[6] http://inspire.ec.europa.eu
[7] https://home.cern/about/updates/2014/03/tool-developed-cern-makes-software-citation-easier
[8] https://www.software.ac.uk/
[9] https://travis-ci.org

Handlungsempfehlungen

Der Aufbau und nachhaltige Betrieb von technischen Infrastrukturen zur kollaborativen Softwareentwicklung und zur dauerhaften Speicherung der wissenschaftlichen Softwareprojekte sollte entsprechend finanziert und gefördert werden. Insbesondere im Bereich der Infrastruktur sind längerfristige Perspektiven für qualifiziertes Personal von essentieller Bedeutung um die Nachhaltigkeit zu gewährleisten (vgl. Handlungsempfehlungen zu Personal, Ausbildung, Karrierewege). Es sollte auf eine Balance zwischen der Entwicklung eigener Infrastrukturen und der Nutzung kommerzieller Infrastrukturen geachtet werden, um Abhängigkeiten zu vermeiden. Bei der Nutzung von externen Cloud-Diensten müssen Schnittstellenfunktionen vorhanden sein, die eine Verknüpfung mit lokalen Repositorien ermöglichen. Nicht nur für die Softwareentwicklung, sondern auch für die technischen Infrastrukturen sind die Anwendung von Qualitätsstandards und Best-Practices-Verfahren von großer Bedeutung. Die Berücksichtigung von Testdaten und -umgebungen als Anforderung an Software-Infrastrukturen sind eine wichtige Maßnahme zur Sicherung von Qualität und Nachhaltigkeit der verwalteten Projekte. Hierzu sollte der Dialog über wissenschaftliche Einrichtungen hinweg gefördert werden.

4. Standards und Qualitätssicherung

Relevanz

Standards und Qualitätssicherung sind elementar für die Zugänglichmachung und Nachnutzung von Software. Die Sicherstellung von Maßnahmen der Qualitätssicherung bilden die Grundlage für die Nachvollziehbarkeit und Nachnutzung von Software. Die Anwendung etablierter Standards trägt zur Interoperabilität des entwickelten Softwarecodes bei.

Fragestellungen

- Was sind Mindeststandards für wissenschaftliche Software und wie lassen sie sich in Leit- und Richtlinien umsetzen?
- Welche Rolle spielen Code-Dokumentation, Nutzerdokumentation und Versionskontrolle?
- Wie kann die Qualitätskontrolle von Software sichergestellt werden?
- Wie wird nachhaltige Softwareentwicklung an Nachwuchswissenschaftler/innen vermittelt?
- Wie wird die Zuständigkeit für die Wartung der Software bei personellen Veränderungen in der Einrichtung geregelt?

Herausforderungen

Aufgrund der hohen Mobilität und Personalfluktuation in der Wissenschaft kommt der Anwendung von Standards und den Maßnahmen der Qualitätssicherung eine besonders hohe Bedeutung zu. Es fehlt an Anreizen dafür, dass Entwickler/innen die Verantwortung für die Qualität der Software übernehmen. Allgemeine Mindestanforderungen sind schwer so zu definieren, dass sie den diversen disziplinären Anwendungen gerecht werden.
In Bezug auf die Qualitätskontrolle gibt es in der Praxis noch viele manuelle Schritte, die zeitaufwendig und fehleranfällig sind. Wenn es um den Code-Review-Prozess geht, sind Sicherheits- und Architekturfragen zu klären, zum Beispiel beim Ausführen von binärem Code.

Lösungsansätze

Die absolute Mindestanforderung umfasst die Dokumentation und Ausführbarkeit von Software. Generelle Regeln der Softwareentwicklung und Verweise auf die Code-Styles der verschiedenen Programmiersprachen können dazu beitragen Standards zu etablieren. Für

den Erfolg von Standards ist die Verankerung im sozialen Kontext und Unterstützung durch Automatisierung von großer Bedeutung. Container-Lösungen (z. B. Docker) ermöglichen die Reproduzierbarkeit der Ausführung der Software in einer definierten Umgebung.
In Bezug auf die Qualitätssicherung ist der Peer-Review-Prozess ein Lösungsansatz, der im Bereich der Publikationen etabliert ist und auch auf Software übertragen werden kann.

Handlungsempfehlungen

In Bezug auf Mindeststandards ist der Austausch von Best-Practices wichtig. Es sollten Style-Guides vorliegen und Verantwortlichkeiten definiert werden. Es sollte eine deutliche Trennung zwischen dem Nötigen und dem Wünschenswerten geben. In Bezug auf den Code-Review-Prozess sollten die Erfahrungen aus dem Review-Prozess bei Publikationen an die Spezifika von Software angepasst werden. Dabei sollten Container-Lösungen für Software genutzt werden. Der Wissenstransfer zwischen den verschiedenen Communities sollte gefördert werden.

5. Reproduzierbarkeit

Relevanz

Reproduzierbarkeit ist die Basis für Vertrauen in wissenschaftliche Ergebnisse. Im Sinne der Reproduzierbarkeit der Ergebnisse computergestützter wissenschaftlicher Arbeit muss Software immer in Verbindung mit der entsprechenden technischen Infrastruktur, den Forschungsdaten und den genutzten Workflows betrachtet werden.

Fragestellungen

- Wie wird Reproduzierbarkeit definiert und welche Rolle spielen disziplinspezifische Unterschiede?
- Wie lange ist die Reproduzierbarkeit von Software notwendig, wünschenswert, und technisch realisierbar?
- Wie kann die Bereitstellung von Software als integraler Bestandteil der wissenschaftlichen Arbeit forciert werden?
- Wie sollten die Funktionen einer Software beschrieben werden, um die Reproduzierbarkeit zu fördern?

Herausforderungen

Es gibt verschiedene Ebenen der Reproduzierbarkeit (z. B. die exakte Reproduktion von Ergebnissen vs. die Reproduktion eines abstrakten Modells), die unterschiedlichen Abhängigkeiten unterliegen (Nutzung von kommerzieller Software vs. Umgang mit selbst entwickeltem Code; Closed vs. Open Source). Eine umfassende Code-Dokumentation ist essentiell für das Verständnis und die Nachnutzung der Software. Zu berücksichtigen sind auch Community-spezifische Problemstellungen, die sich z. B. in den unterschiedlichen Software-Formaten niederschlagen. So müssen je nach Softwaretyp unterschiedliche Strategien zur Gewährleistung von Reproduzierbarkeit verfolgt werden.

Lösungsansätze

Um Ergebnisse reproduzierbar zu machen, muss wissenschaftliche Software offen zugänglich, funktionsfähig und mit eindeutigen Versionsnummern versehen sein. In den Software Communities gibt es etablierte Praktiken zur Versionskontrolle von Software, zur Verwendung von offenen und standardisierten Lizenzen, die eine Nachnutzung der Software erlauben, sowie für die Zugänglichmachung von Software in offen zugänglichen Repositorien. Diese Prinzipien sind auch auf wissenschaftliche Software anwendbar.

Handlungsempfehlungen

Die Reproduzierbarkeit von Software kann durch eine ausführliche Softwaredokumentation gefördert werden. Die Softwaredokumentation sollte deshalb sowohl in Richtlinien zur guten wissenschaftlichen Praxis, als auch in institutionellen Policies zur Veröffentlichung verankert, und in der Aus- und Weiterbildung klar vermittelt werden (vgl. Session Personal, Ausbildung, Karrierewege). Wissenschaftliche Einrichtungen können technische Infrastrukturen sowie Informationsangebote und Service-Personal bereitstellen, um die Veröffentlichung der Software unter offenen und standardisierten Lizenzen nach Grundsätzen der Reproduzierbarkeit zu fördern (vgl. Handlungsempfehlungen zu Lizenzierung und weitere rechtliche Aspekte). Die Veröffentlichung von gut dokumentierter Software sollte gewürdigt und die Dokumentation beispielsweise durch Community-spezifische Beispieldatensätze (Benchmarking) unterstützt werden. Die Relevanz des Themas sollte früh in der Lehre verankert sein, aber auch die kontinuierliche Sensibilisierung bei schon länger in der Wissenschaft tätigen Personen, Arbeitsgruppenleiter/innen, sowie bei Verantwortlichen in Institutionen und Politik sollte stattfinden.

6. Lizenzierung und weitere rechtliche Aspekte

Relevanz

Die Nachnutzung von Software erfordert eine Lizenzierung des Programmcodes. Es gibt ein Spannungsfeld in Bezug auf die Verwertungsrechte: einerseits kann die Möglichkeit des Zugangs zu und die Nachnutzung von wissenschaftlicher Software im Sinne von Open Source das Ansehen der Wissenschaftler/innen stärken und den Austausch von Ideen befördern, andererseits ist die kommerzielle Verwertung von Software für einige Einrichtungen von Interesse. Dieses Spannungsfeld gilt es bei jeder Veröffentlichung abzuwägen.

Fragestellungen

- Welche Lizenzierungen gehen mit welchen rechtlichen Implikationen einher?
- Wie wird das entsprechende rechtliche Know-how Wissenschaftler/innen zur Verfügung gestellt?
- Welche Lizenzmodelle werden mit welchem Ziel bevorzugt und wie werden diese umgesetzt?
- Welche Rolle spielen der offene Zugang, die kommerzielle Verwertung und die Exportkontrolle?

Herausforderungen

Die Kenntnisse über geeignete Lizenzen für Software und ihre Implikationen variieren je nach wissenschaftlicher Einrichtung. Häufig sind die Rechtsabteilungen nicht auskunftsfähig. Entwickler/innen kennen zwar die Lizenzen, treffen jedoch häufig nicht die Entscheidung über die Veröffentlichungs- und Verwertungsstrategie. Die Entscheidung, welche Lizenz verwendet wird, wird in der Praxis oft von den Projektverantwortlichen selbst getroffen. Meistens gibt es keine definierten und dokumentierten Prozesse in Einrichtungen für die Veröffentlichung von Software bzw. keine Kenntnis darüber; nur vereinzelt sind institutsweite Regelungen vorhanden.

Lösungsansätze

Es gibt in einigen Institutionen Informationsbroschüren und Checklisten, sowie Dokumentationen von Kriterien und Best-Practices. Es gibt Beratungsangebote zu Lizenzfragen sowie Weiterbildungsangebote. Bei internationalen Kollaborationen ist die Rechtslage aufgrund von unterschiedlichen Rechtssystemen kompliziert – die Berücksichtigung der Lizenzierung von Software in Kollaborationsverträgen und die

Verwendung von möglichst offenen Lizenzen oder das Verfassen eines „Memorandum of Understanding“ kann helfen, das Thema sachgerecht zu behandeln. Darüber hinaus sind duale Lizenzen und Embargofristen für Software mögliche Lösungsansätze.

Handlungsempfehlungen

Wissenschaftler/innen, aber auch die Leitungsebenen wissenschaftlicher Einrichtungen, sollten bezüglich der Verwertung von wissenschaftlicher Software sensibilisiert werden. Die Zurverfügungstellung von Informationsbroschüren und Checklisten wird empfohlen. Die Publikation von Software sollte einen definierten Genehmigungsworkflow durchlaufen. Die Einschränkung auf wenige mögliche Lizenzen erleichtert es, den Überblick zu wahren. Beratung zu Lizenzfragen an den wissenschaftlichen Einrichtungen sowie die Möglichkeit, externe Beratung durch Spezialist/innen in Anwaltskanzleien einzuholen, sollte durch wissenschaftliche Einrichtungen angeboten werden. Zur Umsetzung dieser Empfehlungen erscheint ein verstärkter Austausch zwischen den Einrichtungen und über die Grenzen der verschiedenen Wissenschaftsorganisationen hinweg sinnvoll.

7. Zitation und Anerkennung

Relevanz

Mit der Forderung nach Open Science gewinnt die Diskussion über geeignete Publikations- und Zitationsstrategien für wissenschaftliche Software an Bedeutung. Die Einhaltung von Zitationsstandards für Software in der Wissenschaftspraxis ist eine wichtige Voraussetzung, um die Veröffentlichung von Software zu fördern.

Fragestellungen

- Was sind Anreize für Forschende, ihren Code zu veröffentlichen?
- Welche Formen von Anerkennung der Leistung, die beim Schreiben von wissenschaftlicher Software erbracht wird, gibt es?
- Welche Rahmenbedingungen sind für die Einführung von zitierbaren Identifikatoren (z. B. dem Digital Object Identifier - DOI) für Software notwendig?
- Wie geht man mit Blick auf die Zitation mit notwendigen Änderungen der Software um?
- Gibt es einen Interessenkonflikt zwischen den Autor/innen einer wissenschaftlichen Software und den zukünftigen Nutzer/innen?
- Wie geht man mit Software um, deren Anwendung zu wirtschaftlich verwertbaren Produkten führen kann/soll?

Herausforderungen

Für Wissenschaftler/innen stellt die exklusive Verwendung des von ihnen entwickelten Codes einen Wettbewerbsvorteil dar. Die Nachnutzung des Codes durch andere Wissenschaftler/innen, die nicht an der Entwicklung des Codes beteiligt waren, kann verschiedene Konsequenzen haben. Die Veröffentlichung des Codes bietet für Wissenschaftler/innen einerseits eine Möglichkeit, durch Zitationen Anerkennung zu gewinnen. Andererseits kann die Verfügbarkeit des Codes den Wettbewerbsvorteil reduzieren. Es gibt noch keine etablierten Publikations- und Zitationsstrategien für wissenschaftliche Software. Die Leistung von Software-Entwickler/innen in der Wissenschaft wird häufig nicht genügend anerkannt.

Lösungsansätze

Die Zitation einer Software[10] erfolgt generell bei einer auf der Software basierten Veröffentlichung oder bei der Nachnutzung des Programmcodes. Die Bereitstellung von Software in einer zitierfähigen Form mit Hilfe von persistenten Identifikatoren, wie z. B. dem Digital Object Identifier (DOI), ermöglicht es, die jeweils aktuelle Version der Software zu referenzieren und diese gegebenenfalls für eine Nachnutzung bereit zu stellen, was eine positive Auswirkung auf den Karriereweg einer/eines Entwickler/in haben kann. Um den Wettbewerbsvorteil der Wissenschaftler/innen, die den Code entwickelt haben, zu wahren, sind bei der Veröffentlichung von Software Einschränkungen des Kreises der Benutzer/innen, z. B. durch Zugang auf Anfrage oder Embargofristen, denkbar.

Handlungsempfehlungen

Wissenschaftliche Software sollte ein eigener Publikationstyp sein, der als eigenständiges, eindeutig identifizierbares, wissenschaftliches Produkt wahrgenommen und geschätzt wird. Dazu müssen gegebenenfalls Formate und Metadaten etabliert werden, in denen der Code mit entsprechenden Anleitungen und Kommentaren beschrieben ist, nicht jedoch die aus dem Code abgeleiteten Ergebnisse. Die Veröffentlichung von wissenschaftlicher Software sollte einen gleichwertigen Stellenwert haben wie die Veröffentlichung wissenschaftlicher Publikationen. Zitationsstandards für wissenschaftliche Software sind notwendig. Persistente Identifikatoren für die Softwareentwicklung, wie z. B. der Digital Object Identifier (DOI), müssen die Versionshistorie mit abbilden und Abhängigkeiten aufzeigen. Das Zitieren von wissenschaftlicher Software und die damit einhergehende Anerkennung sollte bei der Evaluierung von Forschungsleistungen berücksichtigt und im Wissenschaftssystem verankert werden. Auf Publikationslisten sollten neben textuellen Publikationen und Forschungsdaten auch Software-Publikationen genannt werden können. Journals sollten das Zitat einer Software in den Referenzen anerkennen. Softwareentwickler/innen sollten auch bei Publikationen klar in ihrer Rolle benannt und gewürdigt werden. Eine ledigliche Nennung der Softwareentwickler/innen in den Acknowledgements einer Publikation sollte vermieden werden.

Das Interesse von Förderorganisationen an der Messung des Impacts von geförderten Projekten lässt erwarten, dass für wissenschaftliche Software Metriken herausgearbeitet werden. Diese sollten den Bedürfnissen von wissenschaftlicher Software entsprechen und möglichst offen gestaltet sein.

[10] Beispiel: http://depsy.org

8. Sichtbarkeit und Modularität

Relevanz

Die Sichtbarkeit bestehender und neu geschaffener Software ist einerseits wichtig für die Auffindbarkeit und damit für eine nachhaltige Nutzung der Software und die Nachnutzung des Codes. Andererseits ist sie entscheidend für die Anerkennung der bei der Softwareentwicklung erbrachten Leistung und damit für die Karrieren von Wissenschaftler/innen und Softwareentwickler/innen.

Fragestellungen

- Wie wird Software für relevante Wissenschaftsbereiche sichtbar gemacht?
- Wie kann man redundante Entwicklungsarbeit reduzieren?
- Wie kann man die zentrale Rolle von Software für die Wissenschaft innerhalb des Wissenschaftsbetriebs und darüber hinaus (Drittmittelgeber, Politik, Gesellschaft) deutlicher machen?
- Wie kann man die Softwareentwicklung, die zugehörige Dokumentation, die Qualitätssicherung, den Nutzersupport und die langfristige Funktionssicherung aufwerten?
- Welche Rolle spielt die wissenschaftliche Öffentlichkeitsarbeit und das Reporting?
- Wie bekommen Wissenschaftler/innen Wertschätzung für entwickelte Software?
- Welche Rolle spielen Zitationen in Publikationen?
- Welche Rollen spielen neue und etablierte Metriken?

Herausforderungen

Unzureichende Sichtbarkeit von Software ist auf den Mangel an Verzeichnissen oder guter Annotation bezüglich der Nutzung der Software und/oder Nachnutzung des Source Codes zurückzuführen. Das Fehlen einer anschaulichen Dokumentation führt zu langen Einarbeitungszeiten. Die Nutzung von kommerzieller Software erschwert die Modularität. Existierende (modulare) Frameworks sind hingegen häufig zu komplex für die Einarbeitung Einzelner. Wissenschaftler/innen, die Software entwickeln, sind oft Einzelkämpfer/innen in fachspezifischen Arbeitsgruppen mit geringer oder keiner Programmiererfahrung, die bezüglich der Softwareentwicklung kaum vernetzt sind.

Lösungsansätze

Wichtige Elemente einer erfolgreichen Modularisierung sind unter anderen: eine Standardisierung von Datenformaten, die Implementierung und Dokumentation von Programmierschnittstellen (APIs) sowie die Wartung von Frameworks und ihren Erweiterungsschnittstellen. Dies erfordert eine entsprechende Ausbildung von Softwareentwickler/innen und eine Förderung von Strukturen, die den dadurch entstehenden Mehraufwand bei Konzeption, Dokumentation und Wartung von Frameworks tragen und angemessen honorieren.
Im Hinblick auf die Sichtbarkeit von Software und Modulen ermöglichen maschinenlesbare und damit automatisch verknüpfbare Verzeichnisse von Software die Suche in existierenden Softwarebeständen und auch die Integration in Forschungsinformationssysteme.

Beispiel:

- MOSSCO[11]

Handlungsempfehlungen

Bei der Entwicklung von wissenschaftlicher Software sollte Modularität von Anfang an bedacht werden. Eine saubere und anschauliche Dokumentation aus der sowohl Zusammenhänge als auch Abhängigkeiten zu entnehmen sind, ist bei der Planung der Softwareentwicklung zu beachten. Die Verwendung von Standards ist für die Auffindbarkeit, aber auch für die Nachnutzung/Weiterentwicklung von Software-Modulen in anderen Projekten eine Grundlage. Bei der Entwicklung von Standards und der Betreuung modularer Frameworks ist es entscheidend, dass es eine entsprechende zentrale Unterstützung dafür von und für fachspezifische Communities gibt (vgl. Handlungsempfehlungen zu Personal, Ausbildung, Karrierewege). Auch für die Sichtbarkeit und das Auffinden von Softwarelösungen sind Kommunikation und Vernetzung innerhalb der Communities und darüber hinaus bedeutend.

[11] http://www.mossco.de

9. Geschäftsmodelle

Relevanz

Die Forderung nach Zugang zu und Nachnutzung von Software steht im Einklang mit den wissenschaftspolitisch geforderten Verwertungsstrategien. Während die Verwendung offener Lizenzen die Reproduzierbarkeit wissenschaftlicher Resultate unterstützt, können kommerziell orientierte Verwertungsverfahren einen Beitrag zur finanziellen Absicherung der Weiterentwicklung einer Software leisten. Es empfiehlt sich, mögliche Verwertungsstrategien in einem frühen Stadium der Softwareentwicklung zu thematisieren und eine Managementstrategie für Software zu entwickeln, wie sie auch für Publikationen und Forschungsdaten nötig ist.

Fragestellungen

- Wie ist das Zusammenspiel zwischen kommerzieller Software und an wissenschaftlichen Einrichtungen entwickelter Software? Gibt es Beispiele für die Kommerzialisierung von wissenschaftlicher Software? Wie wird damit umgegangen?
- Welche infrastrukturellen Voraussetzungen haben die wissenschaftlichen Institutionen in Bezug auf die Möglichkeit, Kommerzialisierung in konkurrenzfähiger Weise zu betreiben? Wie wird das Zusammenspiel mit quelloffener Software gestaltet?
- Gibt es Beispiele für duale Lizenzierungsmodelle?
- Wie verträgt sich Kommerzialisierung von Software mit dem freien Austausch von Wissen als Grundlage von Wissenschaft?

Herausforderungen

Wissenschaftliche Einrichtungen haben unterschiedliche Perspektiven auf die Entwicklung von Software und korrespondierende Geschäftsmodelle. Auch in den Wissenschaftsdisziplinen wird unterschiedlich mit der Entwicklung von wissenschaftlicher Software umgegangen. Finanzierungsmodelle zur längerfristigen Weiterentwicklung von wissenschaftlicher Software über eine Projektförderung hinaus sind wenig etabliert. Darüber hinaus ist die Identifikation von Nutzer/innen für wissenschaftliche Software – über die Grenzen der Wissenschaft hinaus – eine Herausforderung. Die Anforderungen der Wirtschaft an Software sind häufig sehr unterschiedlich zu den Verfahren in der Wissenschaft. Dadurch sind wissenschaftliche Softwarelösungen oft nicht in beiden Bereichen anwendbar.

Lösungsansätze

Die Entwicklung von Software trägt in vielen Wissenschaftsbereichen zur Lösung wissenschaftlicher Fragestellungen bei. Die frühe Thematisierung möglicher Veröffentlichungs- und Verwertungsstrategien während des Entwicklungsprozesses kann helfen, eine sachgerechte Strategie für den Umgang mit der Software zu entwickeln und Klarheit über die möglichen Geschäftsmodelle zu gewinnen.
Für die Entwicklung von Geschäftsmodellen für wissenschaftliche Software gibt es vielfältige Möglichkeiten, z. B. die Kombination von Open-Source-Software mit kostenpflichtigem Consulting oder alternativer Lizenzierung für kommerzielle Zwecke. Die Veröffentlichung der Software als Open Source kann auch als eine Marketing-Strategie verstanden werden, über die eine wissenschaftliche Einrichtung ihr Know-how sichtbar macht. Auch das Hosting einer Software durch eine wissenschaftliche Einrichtung kann eine Verwertungsstrategie sein und einen Beitrag zur Weiterentwicklung der technischen Infrastruktur leisten.

Beispiel:

- CGAL[12]

Handlungsempfehlungen

Wissenschaftliche Einrichtungen sollten Strategien und Richtlinien für den Umgang mit Software entwickeln, in denen das Spannungsfeld zwischen kommerzieller Verwertung und geforderter Offenheit im Sinne von Open Science thematisiert werden. Jede Software (Programmcode und ausführbare Programme) sollte auf eine mögliche Verwertungsstrategie geprüft werden. Ziel sollte aus wissenschaftlicher Sicht jedoch immer sein, Software so offen wie möglich zugänglich und nachnutzbar zu machen. Hierzu sollten möglichst offene Lizenzmodelle genutzt werden.
Eine Offenheit gegenüber verschiedenen Geschäftsmodellen für wissenschaftliche Software ist begrüßenswert, da sie zur Diversität von im Wissenschaftskontext anwendbaren Softwarelösungen beitragen kann.

[12] http://www.cgal.org

10. Personal, Ausbildung, Karrierewege

Relevanz

Vorbildung und das Lernen in der Praxis sind für hinreichende Kenntnisse in der Softwareentwicklung gleichermaßen entscheidend, genauso wie verschiedene Ausbildungswege nötig sind, im Rahmen derer entsprechende Kenntnisse erworben werden können.
In der aktuellen Situation „hängen“ Softwareentwickler/innen in der Wissenschaft häufig zwischen den Karrierewegen, egal ob sie als Fachwissenschaftler/innen Software entwickeln und die entsprechende Arbeit für die Karriereentwicklung nicht ausreichend honoriert wird oder ob für sie als Softwareentwickler/innen ohne Interesse an einer wissenschaftlichen Karriere schlicht keine relevanten Stellenformate und Karrierewege existieren.

Fragestellungen

- Zu welchem Zeitpunkt sollen Fähigkeiten zur Softwareentwicklung vermittelt werden?
- Welche Ausbildungswege sollte es geben?
- Wann ist der richtige Zeitpunkt um Weiterbildungen anzubieten und wie kann das Gelernte praktisch umgesetzt werden?

Herausforderungen

Der Zeitpunkt und das Format für die Vermittlung von Fähigkeiten, die für die Entwicklung von wissenschaftlicher Software nötig sind, haben unterschiedliche Konsequenzen. Bei einer Fokussierung auf Informatikkenntnisse in der Hochschulausbildung drohen die Fachkenntnisse zu kurz zu kommen. Bei einer Fokussierung auf das Fachwissen hingegen kann es sein, dass die nötigen informatischen Kenntnisse nicht rechtzeitig erlernt werden.
Die Kommunikation zwischen Fachwissenschaftler/innen und Softwareentwickler/innen erfordert die Entwicklung eines gemeinsamen Verständnisses der Problematik.
Softwareentwicklung wird im Wissenschaftsbetrieb kaum als wichtige Leistung anerkannt, sondern häufig als reiner Service begriffen. Es entstehen zwar allmählich neue Stellenformate für Softwareentwickler/innen in der Wissenschaft, allerdings machen niedrige Bezahlung und kurze Verträge Karrierewege für Softwareentwickler/innen in der Wissenschaft (insbesondere in Positionen, die keine wissenschaftliche Tätigkeit im Sinne der Publikation von Erkenntnissen erfordern) unattraktiv.

Lösungsansätze

Die Vermittlung von Fähigkeiten zur Softwareentwicklung kann zu unterschiedlichen Zeitpunkten (Vermittlung von Coding in Schulen[13], Hochschulen, während der Ausbildung, arbeitsbegleitend) in unterschiedlichen Formaten (theoretisch vs. angewandt; formell vs. informell; individuell vs. in Gruppen / Austausch; verordnet vs. initiativ) und auf unterschiedlichen Wegen (Ausbildung, duales Studium, Studium) stattfinden. Konkrete Strukturen und die Stärkung von Kommunikationskompetenzen und Kommunikationswegen zur Einbettung von Softwareentwickler/innen in wissenschaftlichen Arbeitsgruppen sind wichtig.

Beispiele:

- Formate zur Weiterbildung: Hackathons, Praktika, Mentoring, Coaching, Beratung, Hacky Hour, Software-Kaffee, lokale Communities mit Seminarreihen, Stack Exchange, Software-Carpentry-Kurse[14], Software Sustainability Institute[15]
- Duales Studium: Kooperativer Bachelor-Studiengang Informatik (KoSI) an der FH Darmstadt[16]

Handlungsempfehlungen

Im Kontext großer Software-Entwicklungsprojekte sollten dauerhaft finanzierte Stellen für Entwickler/innen geschaffen werden, wodurch die entsprechenden Institutionen nachhaltig sowohl von der entwickelten Software als auch von den durch Entwickler/innen erworbenen Softwareentwicklungs-Fähigkeiten profitieren können. Die Stellen müssen dabei Attraktivität bieten, um qualifiziertes Personal werben und halten zu können. Für eine nachhaltige Entwicklung und Nutzbarmachung der Software ist dabei auf eine gute Verankerung und Vernetzung der Softwareentwickler/innen hinzuwirken. In diesem Sinne sollte die Interdisziplinarität als Prinzip gestärkt werden, um an den wissenschaftlichen Einrichtungen die Zusammenarbeit zwischen Fachwissenschaftler/innen und Softwareentwickler/innen zu verbessern. Über die Einrichtungen hinaus sollte auch eine Vernetzung und Verankerung der Entwickler/innen und damit der von ihnen betreuten Softwareprojekte in den jeweiligen Fach-Communities gezielt gefördert werden (vgl. Handlungsempfehlungen zu Sichtbarkeit und

[13] z.B.: http://calliope.cc/mission, http://codeweek.eu/, http://www.inf-schule.de/
[14] www.software-carpentry.org
[15] https://www.software.ac.uk
[16] https://www.fbi.h-da.de/studium/informatik-dual.html sowie https://www.fbi.h-da.de/studium/informatik-dual/bachelorstudienmodelle/kosi-bsc.html

Modularität).

Bestehende duale Studiengänge sollten gestärkt, ausgebaut und direkt mit Forschungseinrichtungen vernetzt werden, um von vornherein auch Einblicke in die Fachwissenschaften zu ermöglichen und die Kommunikation und die Zusammenarbeit zu fördern. Es sollten neue Ausbildungs- und Karrierewege für „Research Software Engineers“ an Hochschulen und außeruniversitären Einrichtungen geschaffen werden. Fortbildungen und Netzwerke zu Softwareentwicklung können gefördert werden, z. B. indem konkrete Formate in Förderrichtlinien erwähnt oder angeregt werden. Auch die Verleihung von Preisen für nachhaltige Softwareentwicklung könnte eine positive Auswirkung auf die Bedeutung des Themas in der öffentlichen Wahrnehmung haben.

11. Ausblick

Mit der Forderung nach Open Science gewinnt auch die Diskussion über den Zugang zu und die Nachnutzung von wissenschaftlicher Software an Relevanz. Zwar gibt es im Bereich von Open-Source-Software viele etablierte Strategien für die nachhaltige Entwicklung von Software, diese Verfahren sind allerdings nicht ohne Weiteres auf den Wissenschaftsbetrieb übertragbar. In diesem Bereich sind als Besonderheiten z. B. Reputationssysteme, Verwertungsverfahren und die hohe Mobilität der Forschenden zu berücksichtigen. Vor diesem Hintergrund gilt es, Strategien für den Umgang mit wissenschaftlicher Software zu entwickeln.

Bisher ist das Thema in wissenschaftspolitischen Diskussionen und bei Förderinitiativen tendenziell unterrepräsentiert, jedoch sind in diesem Themenfeld auf vielen Ebenen diverse Aktivitäten sowie eindeutiger Diskussions- und Handlungsbedarf zu verzeichnen. In der Helmholtz-Gemeinschaft, aber auch in anderen außeruniversitären Einrichtungen und an den Hochschulen, gewinnt das Thema wissenschaftliche Software zunehmend an Aufmerksamkeit und Bedeutung.

Der Helmholtz Open Science Workshop „Zugang zu und Nachnutzung von wissenschaftlicher Software“ war ein Impuls der Helmholtz-Gemeinschaft, den Austausch und die Vernetzung von Akteuren für gemeinsame Aktivitäten in diesem Themenbereich zu stärken. Die im Rahmen des Workshops eingerichtete Mailingliste[17] soll den Dialog zu diesem Thema fördern.

Um das Thema wissenschaftliche Software weiter voranzutreiben, wurde eine Ad-hoc AG „Wissenschaftliche Software“ der Schwerpunktinitiative „Digitale Information“ der Allianz der deutschen Wissenschaftsorganisationen[18] ins Leben gerufen. Aufgabe dieser Ad-hoc AG ist es, herauszuarbeiten, in welchen Formaten und mit welchen voraussichtlichen Ergebnissen die zentralen Themen bearbeitet werden können.

[17] https://www.listserv.dfn.de/sympa/info/wiss-software
[18] http://www.allianzinitiative.de/handlungsfelder/querschnittsthemen/wissenschaftliche-software/

12. Literaturempfehlungen

Artaza, H., Chue Hong, N., Corpas M., et al. 2016. *Top 10 metrics for life science software good practices*. F1000Research 2016, 5 (ELIXIR):2000. DOI: http://doi.org/10.12688/f1000research.9206.1

Druskat, S. 2016. *A proposal for the measurement and documentation of research software sustainability in interactive metadata repositories*. Available at https://arxiv.org/abs/1608.04529

Hettrick, S. 2016. *Research software sustainability: report on a knowledge exchange workshop*. Available at http://repository.jisc.ac.uk/6332/1/Research_Software_Sustainability_Report_on_KE_Workshop_Feb_2016_FINAL.pdf

Niemeyer K.E., Smith A.M., and Katz D. S. *The challenge and promise of software citation for credit, identification, discovery, and reuse*. arXiv: 1601.04734 [cs.CY], 2016. Available at: http://arxiv.org/abs/1601.04734

Pan, X., Yan, E. & Hua, W. *Disciplinary differences of software use and impact in scientific literature.* Scientometrics (2016). http://doi.org/10.1007/s11192-016-2138-4

Rios F. 2016. *The Pathways of Research Software Preservation: An Educational and Planning Resource for Service Development*. D-Lib Magazine 22. DOI:10.1045/july2016-rios http://www.dlib.org/dlib/july16/rios/07rios.html

Smith A.M., Katz D.S., and Niemeyer K.E. FORCE11 Software citation working group. 2016. *Software citation principles*. PeerJ Preprints 4:e2169v4 DOI: http://doi.org/10.7287/peerj.preprints.2169v4

Umweltbundesamt. 2014. *Dokumentation des Fachgesprächs „Nachhaltige Software“ am 28.11.2014*. Available at http://www.umweltbundesamt.de/sites/default/files/medien/378/publikationen/dokumentation_fachgespraech_nachhaltige_software.pdf

Wilson G., Aruliah D.A., Brown C.T., Chue Hong N.P., Davis M., Guy R.T., et al. 2014. *Best practices for scientific computing*. PLoS Biol 12(1): e1001745. DOI: http://doi.org/10.1371/journal.pbio.1001745

Weitere Informationsquellen:

WSSSPE (Workshop on Sustainable Software for Science: Practice and Experiences) Reports
http://wssspe.researchcomputing.org.uk/

Journal of Open Research Software (JORS)
http://openresearchsoftware.metajnl.com

13. Anhang

13.1 Programm

Zeit	Dienstag, 22.11.2016			
12:30	Ankunft, Registrierung			
13:00	Begrüßung	Dr. Uwe Konrad (Helmholtz-Zentrum Dresden-Rossendorf) Dr. Hans Pfeiffenberger (Alfred-Wegener-Institut Helmholtz-Zentrum für Polar- und Meeresforschung)		
13:15	Keynote	*Wege aus der in-silico-Reproduzierbarkeitskrise* Dr. Johannes Köster (Centrum Wiskunde & Informatica Amsterdam)		
14:00	Impulsvorträge	*Entwurf eines Metadatenrepositoriums zur Erfassung technischer Nachhaltigkeit von Forschungssoftware* Stephan Druskat (Humboldt-Universität zu Berlin) *Qualitätsmanagement von und Infrastruktur für Open Source Software in der Wissenschaft am Beispiel des Medical Imaging Interaction Toolkits (MITK)* Dr. Stefan Kislinskiy, Dr. Caspar Goch (Deutsches Krebsforschungszentrum – DKFZ) *Wissenschaftliche Software pflegen – aber wie?* Prof. Dr. Andreas Zeller (Universität des Saarlandes)		
14:30	Vortrag	*Nachhaltigkeit von Forschungssoftware* Dr. Matthias Katerbow (Deutsche Forschungsgemeinschaft)		
15:00	Kaffeepause			
15:30	Sessions	*Geschäftsmodelle* Dr. Jürgen Fuhrmann	*Reproduzierbarkeit* Dr. Bernadette Fritzsch	*Technische Infrastrukturen* Dr. Uwe Konrad
17:00	Präsentation der Ergebnisse	Dr. Jürgen Fuhrmann (Weierstraß-Institut) Dr. Bernadette Fritzsch (Alfred-Wegener-Institut, Helmholtz-Zentrum für Polar- und Meeresforschung) Dr. Uwe Konrad (Helmholtz-Zentrum Dresden-Rossendorf)		
17:30	Diskussion	Moderation: Dr. Wolfgang zu Castell (Helmholtz Zentrum München Deutsches Forschungszentrum für Gesundheit und Umwelt)		
18:00	Ende	Bustransport zum Stadtzentrum Dresden		
19:30	Abendessen (Selbstzahler-basis)	Restaurant Chiaveri, Dresden		

Zeit	**Mittwoch, 23.11.2016**			
08:00	Bustransport ab Dresden			
08:45	Ankunft			
09:00	Keynote	*Reproduzierbarkeit und Open Science:* *Bestandteile und erste Erfahrungswerte* *- mit besonderem Augenmerk auf Software* Dr. Sünje Dallmeier-Tiessen (CERN)		
09:45	Kaffeepause			
10:00	Sessions	*Lizensierung und andere rechtliche Aspekte* Andreas Schreiber	*Standards und Qualitäts-sicherung* Dr. Wolfgang zu Castell	*Sichtbarkeit und Modularität* Gianna Scharnberg
11:30	Präsentation der Ergebnisse	Andreas Schreiber (Deutsches Zentrum für Luft- und Raumfahrt – DLR) Dr. Wolfgang zu Castell (Helmholtz-Zentrum München Deutsches Forschungszentrum für Gesundheit und Umwelt) Gianna Scharnberg (Universität Duisburg-Essen)		
12:00	Mittagsimbiss			
13:00	Sessions	*Personal, Ausbildung und Karrierewege* David Lähnemann	*Zitation und Anerkennung* Dr. Dirk Steglich	
14:30	Präsentation der Ergebnisse	David Lähnemann (Helmholtz-Zentrum für Infektionsforschung) Dr. Dirk Steglich (Helmholtz-Zentrum Geesthacht, Zentrum für Material- und Küstenforschung)		
15:00	Ausblick	Dr. Uwe Konrad (Helmholtz-Zentrum Dresden-Rossendorf) Dr. Bernadette Fritzsch (Alfred-Wegener-Institut, Helmholtz-Zentrum für Polar- und Meeresforschung)		
15:15	Ende der Veranstaltung	Bustransport zum Stadtzentrum Dresden		

13.2 Abstracts der Keynotes und Impulsvorträge

Wege aus der in-silico-Reproduzierbarkeitskrise

Dr. Johannes Köster (Centrum Wiskunde & Informatica Amsterdam)

Wissenschaftliche Software bildet heute die Grundlage vieler Ergebnisse und Studien in datenintensiven Wissenschaften, insbesondere in der Biologie und Medizin. Während in den vergangenen Jahren großer Aufwand betrieben wurde, um die Reproduzierbarkeit von experimentellen Laborergebnissen sicherzustellen, gibt es kaum Vorgaben zur Veröffentlichung, Distribution, und Qualität von wissenschaftlicher Software. Am Beispiel der Bioinformatik beleuchtet dieser Vortrag die daraus resultierende Reproduzierbarkeitskrise und präsentiert allgemein anwendbare Lösungen auf verschiedenen Ebenen.

Reproduzierbarkeit und Open Science: Bestandteile und erste Erfahrungswerte - mit besonderem Augenmerk auf Software

Dr. Sünje Dallmeier-Tiessen (CERN)

Reproduzierbarkeit von wissenschaftlichen Ergebnissen wird zunehmend in vielen Wissenschaftskreisen und in der Öffentlichkeit diskutiert. Der Zugang zu Forschungsdaten, Workflows oder Arbeitsschritten, sowie zur Dokumentation und der zugrunde liegenden Software sind wichtige Bestandteile dieser Diskussion. Insbesondere in Disziplinen, in denen große Datenmengen, Prozessierungsschritte und Modellberechnungen betrieben werden, wird der Langzeitarchivierung und dem Zugang zu wissenschaftlicher Software besondere Beachtung zugeschrieben.

Am CERN und in der Hochenergiephysik gibt es verschiedene Diskussionen und Lösungsansätze, die zum Teil auf andere Communities übertragbar sind und im Vortrag vorgestellt werden. Ein Beispiel ist CERNVM, was bereits als eine Lösung im Bereich Software Preservation in mehreren LHC-Kollaborationen eingesetzt wird.

Um einen Paradigmenwechsel im Hinblick auf Reproduzierbarkeit und Open Science umsetzen zu können, werden verschiedene Komponenten benötigt. Hierbei nehmen neben technischen Lösungen insbesondere die Anreizmechanismen eine Sonderrolle ein, wenn etablierte Community Workflows und neuere Angebote überzeugend zusammengebracht werden müssen. Dies gilt z. B. für automatisierte Verfahren (mit DOI-Registrierung) über Github.

Am Beispiel von INSPIREHEP illustriert der Vortrag, wie in dieser Community versucht wird, pragmatische Lösungsvorschläge einzubringen, die hoffentlich einige Hürden abbauen werden. INSPIREHEP ist eine offen zugängliche Informationsplattform, die alle relevanten

Informationsressourcen in der Hochenergiephysik aggregiert und insbesondere die Daten- und Softwaresammlung ausbaut, um deren Sichtbarkeit zu erhöhen. Daten- und Softwarepublikationen werden gezählt und auf der Autorenseite der Plattform bereitgestellt. Des weiteren zeigen die Arbeiten am CERN Analysis Preservation Service, wie der wissenschaftliche Ablauf früh mit Werkzeugen unterstützt werden kann, die einen vertrauenswürdigen Zugang zu wissenschaftlichen Analysen und zu den resultierenden Materialien auf lange Sicht fördern, und wie Wissenschaftler/innen von diesem Service profitieren können.

Nachhaltigkeit von Forschungssoftware

Dr. Matthias Katerbow (Deutsche Forschungsgemeinschaft)

In jeder Phase des wissenschaftlichen Arbeitens wird in vielen Disziplinen Forschungssoftware genutzt, zum Beispiel zur Generierung, Verarbeitung, Analyse und Visualisierung von Forschungsdaten. In diesem Sinne sind mit „Forschungssoftware" die eigens zum wissenschaftlichen Erkenntnisgewinn erstellten Software-Anwendungen und Software-Bibliotheken gemeint.

Ein Teil der Forschungssoftware, die aus wissenschaftlichen Projekten hervorgeht, hat enormes Potenzial für eine breite Nutzung, die weit über die ursprüngliche Nutzung in einem einzelnen Forschungsprojekt hinausgeht. In diesem Fall entsteht ein über den eigenen Standort hinausgehender, (meist) disziplinspezifischer, aber nicht mehr projektspezifischer Bedarf für die nutzerorientierte Weiterentwicklung, Pflege, Nutzbarhaltung, Emulation, Verbreitung und Archivierung dieser Forschungssoftware.

In diesem Sinne stellt der Vortrag dar, dass zur Ermöglichung der Nachhaltigkeit von Forschungssoftware, die einem größeren Anwenderkreis zur Verfügung gestellt werden soll, der Aufbau und die Erprobung von Infrastrukturen notwendig sind.

Die Nachhaltigkeit von Forschungssoftware steht aber auch in einem größeren Zusammenhang, gerade wenn es um Nachnutzung und Reproduzierbarkeit geht. Im Vortrag wird daher die Relevanz der Förderung von Forschungssoftware aus den grundlegenden Wissenschaftsprinzipien abgeleitet.

Abschließend werden Möglichkeiten der DFG zur Förderung der Nachhaltigkeit von Forschungssoftware vorgestellt.

Entwurf eines Metadatenrepositoriums zur Erfassung technischer Nachhaltigkeit von Forschungssoftware

Stephan Druskat (Humboldt-Universität zu Berlin)

Die ressourcenbezogene Nachhaltigkeit von Forschungssoftware ist abhängig von ihrer technischen Nachhaltigkeit. Die Ziele letzterer können definiert werden als 1. Sicherung der Existenz der Software, 2. Erhaltung ihres Produktivpotentials, 3. Schaffung und Erhalt ihres Potentials für Weiterentwicklung und Adaption. Der Zugang zu existierender, geeigneter und technisch nachhaltiger Software scheitert häufig an fehlender Auffindbarkeit sowie ungenügender Dokumentation der technischen Nachhaltigkeit. Abhilfe schaffen kann ein Metadatenrepositorium, das neben der Anwendungsdokumentation Maße bereitstellt, die das Potential technischer Nachhaltigkeit einer Software abbilden. Bereits existierende Plattformen tun dies nicht, in ungenügendem Umfang oder nur implizit. Nachhaltigkeitsmaße müssen auf Kriterien basieren, die sich nach ihrer Objektivität und Quantifizierbarkeit auf Grundlage der drei Nachhaltigkeitsziele kategorisieren lassen. Erweiterbare Grundlage für diese Kriterienkataloge bieten u. a. das kriterienbasierte Assessment des SSI und das Projekt CodeMeta. Zusätzlich lassen sich durch Interaktivität, wie bspw. die Dokumentation von Softwaregebrauch durch Verknüpfung mit Veröffentlichungen oder die Erfassung subjektiverer Datenpunkte wie Nutzbarkeit, wertvolle weitere Anhaltspunkte für Nachhaltigkeit gewinnen. Die unterschiedliche Qualität der Metadaten - von subjektiv und nicht-quantifizierbar bis objektiv und quantifizierbar - legt eine nach Härte und Reproduzierbarkeit gestaffelte Messung und Dokumentation über mehrere Maße nah. Damit ein solches Repositorium ein geeigneter Dokumentationsträger für Softwaremanagementpläne darstellt, sind aber mindestens folgende Hürden zu überwinden: 1. Operationalisierbare Definition technischer Nachhaltigkeit; 2. Definition, Katalogisierung und Kategorisierung von Nachhaltigkeitskriterien; 3. Entwurf genauer, reproduzierbarer, manipulationssicherer und nachvollziehbarer Maße; 4. Entwurf von Berechnungsalgorithmen für diese Maße.

Qualitätsmanagement von und Infrastruktur für Open Source Software in der Wissenschaft am Beispiel des Medical Imaging Interaction Toolkits (MITK)

Dr. Stefan Kislinskiy, Dr. Caspar Goch (Deutsches Krebsforschungszentrum (DKFZ)

MITK ist eine modulare Softwareplattform im Bereich der medizinischen Bildverarbeitung. Sie wird weltweit sowohl von Forschern zur Entwicklung eingesetzt, als auch als Anwendung von Medizinern, um die Translation der Forschungsergebnisse in die Klinik zu

beschleunigen. An und mit MITK wird in der Abteilung Medizinische und Biologische Informatik des DKFZ Heidelberg seit 15 Jahren entwickelt. Das Umfeld der Wissenschaft bedingt dabei einige Hürden, die es zu überwinden gilt:

(1) Das Gros der Entwickler setzt sich aus Doktoranden und Masteranden zusammen und unterliegt damit einer hohen Fluktuation

(2) Eine fachliche Ausbildung oder tiefergehende Kenntnisse im Bereich der Softwareentwicklung sind meist nicht gegeben

(3) Die Prioritäten der Entwickler liegen auf der eigenen Forschung

(4) Über die Jahre hat sich das Spektrum der Forschungsthemen im Bereich der medizinischen Bildverarbeitung breit aufgefächert.

Daraus ergeben sich besondere Anforderungen an das Qualitätsmanagement und die Software-Infrastruktur, um - ohne großen Mehraufwand für die entwickelnden Forscher - einen hohen Standard aufrechtzuerhalten. Unsere Werkzeuge reichen dabei von der Code- und Versionsverwaltung, über garantierte Plattformunabhängigkeit, kontinuierliche Integration, Dokumentation und Schulungen, Bug und Feature Tracking, automatisierte Regeln und Abläufe, Lizenzen bis hin zum Projektmanagement im allgemeineren Sinne. Wir stellen unsere etablierten Abläufe und Erfahrungen der letzten 15 Jahre in diesem Kontext vor.

“Wissenschaftliche Software pflegen – aber wie?”

Professor Andreas Zeller (Universität des Saarlandes)

Wissenschaftliche Software entsteht oft ungeplant, ist dann unerwartet erfolgreich, und will auf einmal auch für Dritte gepflegt werden, auch wenn die ursprünglichen Forschungsmittel längst verbraucht sind. Wie lässt sich dies organisieren? In meinem Vortrag illustriere ich das Problem am Beispiel der erfolgreichen Bioinformatik-Software BALL, und zeige, wie etablierte Verfahren der Open-Source-Entwicklung helfen, den damit verbundenen Aufwand im Rahmen zu halten.

13.3 Folien

Die Folien des Workshops sind online verfügbar:

- Druskat, Stephan: Entwurf eines Metadatenrepositoriums zur Erfassung technischer Nachhaltigkeit von Forschungssoftware. http://doi.org/10.5281/zenodo.168383

- Dallmeier-Tiessen, Sünje: Reproduzierbarkeit und Open Science: Bestandteile und erste Erfahrungswerte - mit besonderem Augenmerk auf Software. http://doi.org/10.5281/zenodo.168384

- Goch, Caspar & Kislinskiy, Stefan: Qualitätsmanagement von und Infrastruktur für Open Source Software in der Wissenschaft am Beispiel des Medical Imaging Interaction Toolkits (MITK). http://doi.org/10.5281/zenodo.168385

- Katerbow, Matthias: Nachhaltigkeit von Forschungssoftware. http://doi.org/10.5281/zenodo.168387

- Köster, Johannes: Wege aus der in-silico-Reproduzierbarkeitskrise. http://doi.org/10.5281/zenodo.168388

- Zeller, Andreas: Wissenschaftliche Software pflegen – aber wie? http://doi.org/10.5281/zenodo.168389

Unter dem Hashtag #hgfos16 finden sich auf Twitter Eindrücke des Workshops.

13.4 Liste der Teilnehmer/innen

Nachname	Vorname	Institution
Al-Turany	Mohammad	GSI Helmholtzzentrum für Schwerionenforschung
Bauer	Johann	Max Planck Institut für Biochemie Martinsried
Boy	Thomas	German Centre for Integrative Biodiversity Research (iDiv)
Bruch	Christoph	Helmholtz-Gemeinschaft
Bussmann	Michael	Helmholtz-Zentrum Dresden-Rossendorf (HZDR)
Dallmeier-Tiessen	Sünje	CERN
Diesmann	Markus	Forschungszentrum Jülich
Druskat	Stephan	Humboldt-Universität zu Berlin
Erben-Russ	Michael	Fraunhofer-Gesellschaft
Faber	Claas	GEOMAR Helmholtz-Zentrum für Ozeanforschung Kiel
Feuchter	Dirk	Karlsruher Institut für Technologie
Feulner	Georg	Potsdam-Institut für Klimafolgenforschung
Finke	Ants	Helmholtz-Zentrum Berlin für Materialien und Energie
Förstner	Konrad	Universität Würzburg
Franke	Michael	Max Planck Digital Library
Fritzsch	Bernadette	Alfred-Wegener-Institut Helmholtz-Zentrum für Polar- und Meeresforschung
Fuhrmann	Jürgen	Weierstrass-Institut für Angewandte Analysis und Stochastik in der Leibniz-Gemeinschaft
Gerlach	Roman	Friedrich-Schiller-Universität Jena
Goch	Caspar	Deutsches Krebsforschungszentrum
Haas	Holger	Deutsches Krebsforschungszentrum
Haase	Robert	Max-Planck-Institut für Molekulare Zellbiologie und Genetik (MPI-CBG)
Hebing	Marcel	DIW Berlin - Deutsches Institut für Wirtschaftsforschung
Heuveline	Vincent	Universität Heidelberg
Himpe	Christian	Max-Planck-Institut Magdeburg
Hübsch	Gerald	Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden (SLUB)
Huebl	Axel	Helmholtz-Zentrum Dresden-Rossendorf (HZDR)
Janosch	Stephan	Max-Planck-Institut für Molekulare Zellbiologie und Genetik (MPI-CBG)
Jordan	Alexander	Helmholtz-Zentrum Potsdam - Deutsches GeoForschungsZentrum GFZ
Juckeland	Guido	Helmholtz-Zentrum Dresden-Rossendorf (HZDR)

Jung	Reiner	Christian-Albrechts-Universität zu Kiel
Katerbow	Matthias	Deutsche Forschungsgemeinschaft
Kern	Roman	Know-Center
Kislinskiy	Stefan	Deutsches Krebsforschungszentrum
Köhler	Martin	Max Planck Institute for Dynamics of Complex Technical Systems
Konrad	Uwe	Helmholtz-Zentrum Dresden-Rossendorf (HZDR)
Kornmesser	Robert	Helmholtz-Zentrum Potsdam - Deutsches GeoForschungsZentrum GFZ
Köster	Johannes	Centrum Wiskunde & Informatica Amsterdam
Lähnemann	David	Helmholtz-Zentrum für Infektionsforschung
Löwe	Peter	Technische Informationsbibliothek Hannover
Michel	Julia	Max-Born-Institut Berlin, Laserlab-Europe
Mikolaj	Michal	Helmholtz-Zentrum Potsdam - Deutsches GeoForschungsZentrum GFZ
Nsonga	Baldwin	Universität Leipzig
Pampel	Heinz	Helmholtz-Gemeinschaft
Piccioni Koch	Daniela	Karlsruher Institut für Technologie
Radzinski	Katja	Helmholtz-Zentrum Potsdam - Deutsches GeoForschungsZentrum GFZ
Ribbrock	Dirk	Technische Universität Dortmund
Saak	Jens	Max Planck Institute for Dynamics of Complex Technical Systems
Schall	Ralf	Helmholtz-Zentrum Geesthacht Zentrum für Material- und Küstenforschung
Scharnberg	Gianna	Universität Duisburg-Essen
Scheliga	Kaja	Helmholtz-Gemeinschaft
Schirlitz	Frank	Technische Universität Dresden
Schirnick	Carsten	GEOMAR Helmholtz-Zentrum für Ozeanforschung Kiel
Schlauch	Tobias	Deutsches Zentrum für Luft- und Raumfahrt
Schreiber	Andreas	Deutsches Zentrum für Luft- und Raumfahrt
Schröder-Barkhausen	Wolf	Max-Delbrück-Centrum für Molekulare Medizin in der Helmholtz-Gemeinschaft
Schulz	Henrik	Helmholtz-Zentrum Dresden-Rossendorf (HZDR)
Schwennsen	Florian	Deutsches Elektronen-Synchrotron DESY
Skripcak	Tomas	German Cancer Consortium (DKTK)
Steglich	Dirk	Helmholtz-Zentrum Geesthacht Zentrum für Material- und

		Küstenforschung
Steinbach	Peter	Max-Planck-Institut für Molekulare Zellbiologie und Genetik (MPI-CBG)
Struck	Alexander	Cluster of Excellence -- Image Knowledge Gestaltung
Wachsmann	Alf	Max-Delbrück-Centrum für Molekulare Medizin in der Helmholtz-Gemeinschaft
Wieser	Thomas	Helmholtz-Zentrum für Umweltforschung - UFZ
Zacharias	Malte	Helmholtz-Zentrum Dresden-Rossendorf (HZDR)
Zeller	Andreas	Universität des Saarlandes
Zirkelbach	Christian	Christian-Albrechts-Universität zu Kiel
zu Castell	Wolfgang	Helmholtz Zentrum München - Deutsches Forschungszentrum für Gesundheit und Umwelt