

Versionskontrolle mit Git + Kollaboratives Arbeiten im Web mit GitHub

Druskat, Stephan

stephan.druskat@hu-berlin.de

Humboldt-Universität zu Berlin, Deutschland

Rockenberger, Annika

annika.rockenberger@nb.no

Nasjonalbiblioteket, Oslo, Norwegen

Einführung

In den Digital Humanities nehmen Softwareentwicklung und Datenverarbeitung eine zentrale Rolle ein, aber auch über diese Felder hinaus ist kollaboratives, verteiltes, digitales Arbeiten inzwischen ein fester Bestandteil des Forschungsprozesses. Um die Qualität ebenso wie die Optimierung des wissenschaftlichen Erkenntnisprozesses sicherstellen und Nachnutzbarkeit der entstehenden Ressourcen gewährleisten zu können, benötigen Forschende Kenntnisse im Bereich grundlegender digitaler Methoden. Derzeit ist eine Einbettung entsprechender methodischer Ausbildung in die Curricula noch nicht überall implementiert, was zum Teil unzureichende Kenntnisse und daraus folgende Unsicherheit in der Anwendung digitaler Methoden bis über die Erfahrungsstufe der Early Career Researchers hinaus zur Folge hat. Zumindest kurzfristig Abhilfe schaffen können entkoppelte Lehrgänge, die Forschenden grundsätzliche Kenntnisse in digitalen Methoden vermitteln. In diesem Bereich haben sich insbesondere die Lehrpläne der Carpentries (<https://carpentries.org/>) als forschungsbasierte, ergebnis- und lerner*innenorientierte Mittel zum Enablement von Forschungsgemeinschaften hervor getan. Diese fokussieren auf kleine Lernschritte, formative Lernzielüberprüfungen, mentale Modelle und kurze Feedbackzyklen, sowie praxisorientierte Übungen. Zusätzlich stellt die auf Freiwilligenarbeit und Inklusion fokussierte Community der Carpentries einen Multiplikator dar, der eine höhere Durchsetzung der Disziplinen mit den notwendigen Grundkenntnissen im Bereich computergestützter Methoden möglich macht. Daher verfolgen wir im Rahmen zweier halbtägiger Workshops zwei Ziele:

1. Vermittlung von zur Einhaltung guter wissenschaftlicher Praxis im Sinne von Reproduzierbarkeit von Forschungsergebnissen und Open Science notwendigen Grundkenntnissen in der Versionsverwaltung mit Git und dem kollaborativen Arbeiten im Web mit GitHub.

2. Einführung in gemeinschaftlich entwickelte, offene Lehrmethoden auf Grundlage der Carpentries, hier von Software Carpentry (Wilson 2006).

Die Workshops sind thematisch eng miteinander verwandt und lassen sich daher gut kombiniert besuchen, fokussieren aber jeweils verschiedene abgeschlossene Aspekte des Bereichs Kollaboration via Versionskontrolle, so dass auch der Besuch eines der beiden Teilworkshops möglich und sinnvoll ist.

Teilworkshop „Versionskontrolle mit Git“

Der halbtägige Workshop „Versionskontrolle mit Git“ richtet sich an Neulinge im Bereich Versionskontrolle. Vermittelt werden Grundkenntnisse der Arbeit mit dem Versionskontrollsystem Git (Chacon & Straub 2014). Git, ursprünglich entwickelt für die kollaborative Zusammenarbeit am Linux-Kernel, ist nicht beschränkt auf die Versionierung von Software-Quellcode, sondern kann – dank seiner zeilenbasierten Arbeitsweise – auch für andere Dokumenttypen gewinnbringend eingesetzt werden: Manuskripte, Forschungsdaten in nicht-binären Formaten, etc.

Versionskontrollsysteme (VCS) schützen vor Verlust von Arbeitsergebnissen und erlauben einfaches Zurückspulen auf vorangegangene Stände. Zudem erlauben Sie – bei Einsatz in verteilten Arbeitsgruppen – im Rückblick die Zuweisung bestimmter Änderungen zu Mitarbeitenden und ermöglichen so einfache Klärung von Nachfragen. Schließlich weist das VCS zuverlässig auf konfligierende Änderungen hin, was bei anderen Formen der verteilten Kollaboration nicht immer der Fall ist.

Auch einzelnen Forschenden bietet der Einsatz von VCS, und insbesondere Git, Vorteile, wie etwa Commitbeschreibungen, die den Zugang zu auch zeitlich weit zurückliegenden Änderungen und Projektständen erleichtern.

Git oder ein anderes VCS kommt heutzutage in allen großen Softwareprojekten zur Verwendung und wird von den Entwickelnden auch für kleine Projekte genutzt.

Um ein wirklich fundiertes Grundwissen über Git vermitteln zu können verzichtet dieser Teilworkshop auf den Einsatz von grafischen Benutzeroberflächen für Git. Stattdessen kommt die Kommandozeilenimplementierung zum Einsatz, die für alle wichtigen Betriebssysteme erhältlich ist. Um ein komplikationsfreies Arbeiten aller Teilnehmenden auf der Kommandozeile zu ermöglichen werden daher entsprechende Kenntnisse vor Workshopbeginn abgefragt und bei Bedarf zu Beginn des Workshops zielgerichtet aufgebaut.

Im Anschluss werden VCS, ihre Funktionsweisen und Zielstellungen erläutert und die Einrichtung von Git vorgenommen. Es folgen die Vermittlung der grundlegenden Funktionen: Änderungsverfolgung, Exploration der Änderungshistorie, Ausschluss

bestimmter Objekte von der Änderungsverfolgung, Änderungskonflikte und ihre Auflösung und die Arbeit mit entfernten Repositorien.

Darüber hinaus gehende Themen, die von der Versionskontrolle mit Git berührt werden finden ebenfalls Beachtung: Workflows für kollaboratives Arbeiten, Open Science, Lizenzierung, Zitierung und das Bereitstellen eigener Git-Instanzen. Nach Möglichkeit wird abschließend auf GUI-unterstützte Versionierung und Markdown-Formate eingegangen. Diese und vorhergehende Themen können von Teilnehmenden im zweiten Teilworkshop vertieft werden.

Der Teilworkshop beruht auf der entsprechenden Unterrichtseinheit „Version Control with Git“ (Ahmadia et al. 2016) aus dem Curriculum von Software Carpentry. Die Workshopleiter*innen sind zertifizierte Software Carpentry Instructors und es kommen die für das Curriculum entwickelten Methoden zum Einsatz, die es den Teilnehmenden ermöglichen, sich das Gelernte durch angeleitete Übungen und häufige formative Evaluationen aktiv anzueignen.

Teilworkshop „Kollaboratives Arbeiten im Web mit GitHub“

Dieser Teilworkshop führt ein in die Möglichkeiten der Nutzung von GitHub als Plattform für kollaboratives, verteiltes Arbeiten im Web. Komplexer werdende digitalisierte Forschungsprozesse und Projektstrukturen fordern zunehmend kollaboratives Arbeiten. Dabei stellen „Plain Text“-Formate ein problemlos nachnutzbares Mittel der Wahl dar. GitHub ist die derzeit wichtigste Codeplattform und bietet neben Git-Repositorien und eigenen Workflows für das Versionskontrollsystem Git weitere Features, die ein solches Arbeiten ermöglichen und erleichtern. Obgleich die Kernfunktionen von GitHub vornehmlich für die Verwaltung von Softwareprojekten genutzt wird, können diese ebenso gut für andere digitale Objekte, wie Forschungsdaten oder Manuskripte, genutzt werden.

Obwohl GitHub das Git schon im Namen trägt ist es möglich, die kollaborativen Funktionen der Plattform auch ausschließlich über die Weboberfläche von GitHub zu nutzen. Auf diese Weise kann nicht nur Versionskontrolle ausgeübt werden, es können in wenigen Schritten auch Websites erstellt und bereitgestellt werden. All dies, ohne auf die Kommandozeile zurückgreifen zu müssen.

Für den Workshop sind keine Vorkenntnisse über Versionskontrollsysteme, Git oder GitHub notwendig. Teilnehmende des ersten Teilworkshops werden jedoch grundsätzliche Prinzipien der Versionierung wiedererkennen und vertiefen können, beziehungsweise GUI-gestützte Umsetzungen kennenlernen. Die Teilnehmenden lernen, wie man Repositorien einrichtet, Dateien unter Versionskontrolle stellt, direkt mit Anderen an diesen Dateien zusammenarbeitet, beispielsweise

durch Issues und Pull Requests. Auch weiterführende Themen wie Konfliktresolution, die Verwendung von Änderungsvorschlägen und Review und das Aufsetzen einer Website sind Themen des Workshops.

Auch dieser Teilworkshop beruht auf den Methoden der Software Carpentry.

Zusammenfassend ermöglichen die beiden Teilworkshops vor allem in Kombination den Teilnehmenden einen fundierten Einstieg sowohl in die Versionskontrolle allgemein, als auch in die praktische Arbeit mit den derzeit am weitesten verbreiteten Instrumenten, Git und GitHub. Die vermittelten Kenntnisse erlauben es, die Vorteile der Arbeit mit diesen von Anfang an auszunutzen und sind Grundlage für den täglichen Einsatz von Versionskontrolle und kollaborativen Prozessen. Beide sind ein zentraler Stützpfeiler für eine auf Best Practices basierende Forschungsarbeit in den digitalen Geisteswissenschaften und eine Voraussetzung von Open Science.

Workshopleiter*innen

Stephan Druskat, Humboldt-Universität zu Berlin, Institut für deutsche Sprache und Linguistik, Unter den Linden 6, 10099 Berlin. E-Mail: stephan.druskat@hu-berlin.de.

Stephan Druskat ist Magister der Anglistik, Linguistik und neueren deutschen Literatur und arbeitet seit 2009 als Research Software Engineer (RSE) in Forschungsprojekten der Linguistik und der digitalen Geisteswissenschaften. Er ist Co-Convener der Dhd-AG „Research Software Engineering“, Mitbegründer und Vorstandsmitglied von *de-RSE e.V. – Gesellschaft für Forschungssoftware* und Special Collaborator des britischen *Software Sustainability Institute*. In seiner Forschung beschäftigt er sich vor allem mit Nachhaltigkeit von Forschungssoftware und Softwarezitierung.

Annika Rockenberger, Ph.D., Research Librarian for Digital Humanities, The National Library of Norway, Postboks 2674 Solli, NO-0203 Oslo, Norwegen. E-Mail: Annika.Rockenberger@nb.no.

Annika Rockenberger hat in Berlin u.a. Literaturwissenschaft, Geschichte und Kommunikationswissenschaft studiert. Mit einer Arbeit zur Analytischen Philosophie der Editionsphilologie ist sie an der Universität Oslo promoviert worden. Programmieren hat sie sich selbst beigebracht. Sie ist DH Aktivistin und Community Builder in Norwegen, dem skandinavischen Raum und Europa und derzeit für die Entwicklung einer DH Strategie an der norwegischen Nationalbibliothek zuständig.

Bibliographie

Chacon, Scott/ Straub, Ben (2014): *Pro Git (2nd Edition)*. New York City, NY, USA: Apress. 10.1007/978-1-4842-0076-6 .

Ahmadia, Aron / Allen, James / Appling, Alison / Aubin, Sean / Bachant, Pete / Banaszkiewicz, Piotr / Barmby, Pauline / Batut, Berenice / Bekolay, Trevor / Blischak, John / Bonsma, Madeleine / Borrelli, Jon / Boughton, Andy / Bouquin, Daina / Brauning, Rudi / Brett, Matthew / Brown, Amy / Cabunoc, Abigail / Charlesworth, Jane / Charlton, Billy / Chen, Daniel / Christensen, Garret / Collings, Ruth / Corvellec, Marianne / Davis, Matt / Dolson, Emily / Duchesne, Laurent / Duckles, Jonah / Emonet, Rémi / Estève, Loïc / Farsarakis, Emmanouil / Fauber, Bennet / Fouilloux, Anne / Förstner, Konrad / Geiger, Stuart / Gonzalez, Ivan / Guarinello, Marisa / Hadwin, Jamie / Hannah, Nicholas / Hansen, Michael / Heroux, Martin / Hertweck, Kate / Hinsén, Konrad / Huang, Daisie / Ismiraldi, Yuandra / Jackson, Mike / Jacobs, Christian / Jarecka, Dorota / Johnston, Luke W. / Jones, David / J#drzejewski-Szmek, Zbigniew / King, W. Trevor / Kluyver, Thomas / Konrad, Bernhard / Kuzak, Mateusz / Labrie, Kathleen / Lapp, Hilmar / Latornell, Doug / Lauferweiler, Mark / LeBauer, David / Lee, Kate / Liffers, Matthias / Loucks, Catrina / Ma, Keith / Marwaha, Kunal / Michonneau, François / Mills, Bill / Mueller, Andreas / Nagraj, VP / Nederbragt, Lex / Nunez-Iglesias, Juan / O'Brien, Brenna / O'Leary, Aaron / Olsson, Catherine / Pawsey, Chris / Pfenninger, Stefan / Pipitone, Jon / Poisot, Timothée / Preney, Paul / Rice, Timothy / Riemer, Kristina / Rio Deiros, David / Robinson, Natalie / Rohl, Andrew / Rokem, Ariel / Sarahan, Michael / Schmeier, Sebastian / Schmider, Hartmut / Silva, Raniere / Smithyman, Brendan / Soranzo, Nicola / Steinbach, Peter / Stevens, Sarah / Timbers, Tiffany / Traphagen, Danielle / Tröndle, Tim / van der Walt, Anelda / Vandervalk, Steve / Weaver, Belinda / Wheelhouse, Mark / White, Ethan / Wilson, Greg / Wu, Steven / Zhang, Qingpeng (2016): *Software Carpentry: Version Control with Git (Version 2016.06)*, in: Huang, Daisie / Gonzalez, Ivan (Hrsg.): <https://github.com/swcarpentry/git-novice> [letzter Zugriff 13. Oktober 2018]. 10.5281/zenodo.57467 .

Wilson, Greg (2006): *Software Carpentry: Getting Scientists to Write Better Code by Making Them More Productive*, in: *Computing in Science & Engineering* 8: 66-69. 10.1109/MCSE.2006.122 .