Offen für Professionalisierung? Wie Software und Entwickler*innen in den Digital Humanities gestärkt werden können

Czmiel, Alexander
czmiel@bbaw.de
Berlin-Brandenburgische Akademie der Wissenschaften

Henny-Krahmer, Ulrike
ulrike.henny-krahmer@uni-rostock.de
Universität Rostock

Jettka, Daniel
daniel.jettka@uni-paderborn.de
Universität Paderborn

Hintergrund

Forschungssoftware zu entwickeln ist neben der Anwendung digitaler Methoden ein wichtiger Teil der Forschungstätigkeiten in den Digital Humanities, findet jedoch meist immer noch in einer wenig professionalisierten Form statt (siehe schon Schrade et al. 2018). Zwar ist es in manchen Bereichen der DH möglich, für bestimmte Aufgaben auf bestehende, außerhalb der DH selbst entwickelte und teilweise kommerzielle Softwarelösungen zurückzugreifen, z. B. auf den Oxygen XML-Editor (SyncRO Soft SRL 2022) für die Bearbeitung von XML-Dateien oder auf die Software Gephi (Bastian et al. 2009) zur Visualisierung von Netzwerkdaten. Häufig sind aber solche allgemeinen Tools für die geisteswissenschaftlichen Forschungsfragen und -projekte nicht geeignet und speziellere Werkzeuge nicht vorhanden. Das führt dazu, dass die digitalen Geisteswissenschaftler*innen selbst als Software-Entwickler*innen tätig werden oder eng mit Research Software Engineers (RSE) zusammenarbeiten, um neue Forschungssoftware zu entwickeln.

Ein typischer Fall sind digitale Editionen, die einerseits auf einem zugrunde liegenden Datenmodell und einer Repräsentation in Daten – den Forschungsdaten selbst – basieren. Andererseits nehmen sie in den meisten Fällen erst durch eine digitale Webpräsentation Gestalt an, die oft je nach Gegenstand der Edition maßgeschneidert entwickelt wird. Auch für Analysen von geisteswissenschaftlichen Daten kann zwar häufig auf bestehende Softwarekomponenten und -module zurückgegriffen werden, diese werden aber oft erst dadurch sinnvoll für die Forschung nutzbar, dass sie programmtechnisch für einen spezifischen Workflow eingesetzt und miteinander verknüpft werden, wie z. B. einzelne Python-Module für einen Textanalyseworkflow. Oder es werden ganz neue Analysetools für geisteswissenschaftliche Anwendungsfälle entwickelt, wie etwa im Fall des Tools Stylo (Eder et al. 2016) für stilometrische Analysen. Schließlich beinhaltet geisteswissenschaftliche Softwareentwicklung auch das Entwickeln von Skripten zur Aufbereitung und Umwandlung von Daten, damit diese wiederum in anderen Tools verarbeitet werden können.

Art und Umfang der Softwareentwicklung in den Geisteswissenschaften können also stark variieren. Genauso kann auch der Anteil und Stellenwert, den die Tätigkeit „Softwareentwicklung“ für Forscher*innen in den DH hat, sehr unterschiedlich sein. Das Entwickeln von Forschungssoftware kann einen Teil der gesamten Tätigkeit ausmachen, z. B. wenn ein Projekt individuell von Einzelnen umgesetzt wird, von einer geisteswissenschaftlichen Fragestellung über die Operationalisierung und technische Implementierung bis hin zur Interpretation und Aufbereitung der Ergebnisse. Softwareentwicklung kann aber auch für einzelne Forschende zur Haupttätigkeit werden, insbesondere in größeren Teams, die arbeitsteilig tätig sind, oder wenn die Entwicklung von Forschungssoftware im Mittelpunkt eines Vorhabens steht. Für diejenigen, die hauptsächlich mit der Entwicklung von Forschungssoftware beschäftigt sind, hat sich der Begriff *Research Software Engineer* etabliert.

Gerade dadurch, dass die Softwareentwicklung im geisteswissenschaftlichen Forschungskontext und in den DH in den wenigsten Fällen von hierfür ausgebildeten Softwareentwickler*innen oder Informatiker*innen durchgeführt wird, gibt es in diesem Bereich noch viel Raum und viele Möglichkeiten für eine stärkere Professionalisierung. Häufig entsteht geisteswissenschaftliche Forschungssoftware in den ersten Stadien aus der jeweiligen wissenschaftlichen Domäne heraus. Die Forschenden selbst eignen sich hierfür durch Kurse oder Zusatzausbildungen die Grundlagen einzelner Programmiersprachen an. Für diese zusätzliche Kompetenz und auch die zusätzliche Arbeit, die durch die Softwareentwicklung entsteht, gibt es allerdings aus der Community nur selten die verdiente wissenschaftliche Anerkennung. Da weder das Wissen um eine stabile Softwarearchitektur noch ausreichend Zeit für Tests oder gar eine umfangreiche Dokumentation vorhanden ist, ist auch die Software selbst oft in einem verbesserungswürdigen Zustand und nicht für einen nachhaltigen Einsatz vorbereitet. Hier setzt dieser Workshop an.

Ziele des Workshops

Der Workshop wird von der AG „Research Software Engineering in den Digital Humanities“ des DHd-Verbands ausgerichtet und richtet sich an alle Wissenschaftler*innen, die im Rahmen ihrer Forschung oder in Forschungsprojekten Software entwickeln oder an der Entwicklung von Software beteiligt sind. Wir argumentieren, dass eine Offenheit für stärkere Professionalisierung seitens aller

an geisteswissenschaftlicher Softwareentwicklung Beteiligten helfen kann, sowohl die Software selbst zu verbessern als auch die Position ihrer Entwickler*innen in der Wissenschaft zu stärken. Auch wenn Professionalisierung mit steigender Bedeutung von Softwareentwicklung im jeweiligen Kontext im Forschungsprozess und als wissenschaftliche Tätigkeit generell wichtiger wird, können doch alle Bereiche der DH, in denen Softwareentwicklung betrieben wird, von einer stärkeren Professionalisierung profitieren, die grob auf drei Ebenen beschrieben werden kann:

1. technische Professionalisierung

Die technische Professionalisierung betrifft in erster Linie die Bereiche, die in der täglichen Arbeit in der Softwareentwicklung auf der Ebene von Code und Architektur von Bedeutung sind, z. B. *Clean Code* (Martin 2009, Clean Code Developer 2022), Versionierung, Dokumentation, Tests, *DevOps* (Halstenberg et al. 2020) usw.

2. organisatorische Professionalisierung

Die organisatorische Professionalisierung bezieht sich auf alle Aspekte der Planung, des Projektmanagements und der (Selbst-)Organisation von Softwareentwickler*innen. Dazu gehören u. a. die Bildung von Entwickler*innen-Teams, die Verwendung von Versionsverwaltungs- und Ticketsystemen, Methoden der agilen Softwareentwicklung (Beck et al. 2001), aber auch Softwaremanagementpläne (SMPs).

3. institutionelle Professionalisierung

Die institutionelle Professionalisierung schafft die Rahmenbedingungen für die beiden anderen Bereiche, indem Softwareentwicklung Teil von DH-Curricula und die Ausbildung für RSEs in den DH verbessert wird, aber auch Karrierewege ermöglicht werden. Es geht um die Bereitstellung von Mitteln und Infrastruktur für die Entwicklung von Software und die Verankerung der Tätigkeit einer/s RSE im wissenschaftlichen Bereich.

Ausgehend von den drei beschriebenen Ebenen soll der Workshop über mögliche Bereiche der Professionalisierung informieren und einen Erfahrungsaustausch und Diskussionen zum Thema ermöglichen. Konkretes Ziel des Workshops ist es, durch die Beteiligung aller Teilnehmerinnen und Teilnehmer in Gruppenarbeit ein White Paper zu erarbeiten, in dem Professionalisierungsoptionen für die Forschungssoftwareentwicklung in den DH festgehalten werden, so wie sich z. B. auch der Verein de-RSE fächerübergreifend für nachhaltige Softwareentwicklung ausgesprochen hat (Anzt et al. 2020).

Der Workshop soll so dazu beizutragen, dass in den DH mehr nachhaltige Forschungssoftware von hoher Qualität entsteht. Auf allen drei Ebenen - technisch, organisatorisch und institutionell - spielen hierbei auch Aspekte der Offenheit (Open Source, Open Access, Open Science) eine große Rolle, um eine vollständige Integrität und Transparenz aller Prozesse und Werkzeuge im Forschungszyklus zu gewährleisten und eine „gute wissenschaftliche Praxis“ zu ermöglichen. Mehr Professionalität in der Softwareentwicklung sorgt damit für eine bessere Forschungsunterstützung und bessere Forschung in den Geisteswissenschaften, gemäß dem Motto „Better Software, Better Research“ (Goble 2014). Zugleich trägt sie dazu bei, dass Softwareentwicklung als Forschungstätigkeit stärker sichtbar und anerkannt wird, was die Position von Entwickler*innen in den DH verbessert.

Ablauf und Inhalte des Workshops

Der Workshop ist für zwei halbe Tage geplant und setzt auf eine starke Mitarbeit aller Teilnehmenden. Als Ergebnis des Workshops ist angestrebt, ein White Paper mit Best Practices und Professionalisierungsoptionen und -schritten zu publizieren. Das Papier soll als Argumentationshilfe für DH-RSEs dienen, die sich eine professionellere Arbeitsumgebung innerhalb ihrer Institution wünschen. Zudem soll es die Institutionen selbst dabei unterstützen, die Professionalisierung der Forschungssoftwareentwicklung in den DH voranzubringen.

Im ersten Teil des Workshops werden von den Convenern der AG zur inhaltlichen Einführung in das Thema Kurzvorträge zu den Ebenen technischer, organisatorischer und institutioneller Professionalisierung gehalten. Daran schließt sich eine offene Gesprächsrunde mit allen Teilnehmenden an, in der jede und jeder Gelegenheit bekommen soll, auf folgende Fragen einzugehen:

- Wie sehr sehen Sie sich als RSE?
- Wie genau sieht Ihre Rolle als RSE aus?
- Wie schätzen Sie Ihren zukünftigen Karriereweg ein?
- Wie würden Sie Ihren eigenen Professionalisierungsgrad und den Ihrer Institution in Bezug auf Softwareentwicklung einschätzen?
- Halten Sie eine stärkere Professionalisierung in der Softwareentwicklung in den DH für sinnvoll und warum/warum nicht?
- Welche Aspekte halten Sie für eine professionelle Softwareentwicklung für besonders wichtig?

Nach der offenen Gesprächsrunde folgt im zweiten Teil eine gemeinsame Arbeit in Teilgruppen, wobei sich jede Gruppe auf einen der drei Hauptbereiche für Professionalisierung konzentrieren wird. Die Arbeit in den Teilgruppen wird gemeinsam geplant, indem definiert wird, woran jede Gruppe arbeiten und was als Ergebnis der Gruppenarbeit erwartet wird. Für die Dokumentation der Ergebnisse werden Online-Dokumente vorbereitet, in denen kollaborativ gearbeitet werden kann. Jede Teilgruppe wird von einer der einreichenden Personen geleitet. Auch kann jede/r Teilnehmende entscheiden, in welcher Gruppe er oder sie mitarbeiten möchte. Um eine produktive Arbeit sowohl in der Gesamtgruppe als auch in den Teilgruppen zu ermöglichen, ist die maximale Teilnehmerzahl auf 30 Personen begrenzt. Im Folgenden wird ein Überblick über das Programm gegeben:

Teil 1:

- Begrüßung (10min)
- Input-Statements von Seiten der AG zu Teilaspekten von Professionalisierung (30min)
- Offene Gesprächsrunde zu Professionalisierung - Teil I (80min)
- Pause (30min)
- Offene Gesprächsrunde zu Professionalisierung - Teil II (60min)
- Planung und Abstimmung zu Gruppenarbeit (30min)

Teil 2:

- Gruppenarbeit zu Teilaspekten I (90min)

- Pause (30min)
- Gruppenarbeit zu Teilaspekten II (60min)
- Ergebniszusammenführung (30 min)
- Diskussion und Abschluss (30 min)

Zur Durchführung des Workshops werden ein Beamer, ausreichend Steckdosen und WLAN benötigt.

Kontaktdaten der Beitragenden

Alexander Czmiel ist Leiter von TELOTA - IT/DH, der IT und Digital Humanities-Abteilung der Berlin-Brandenburgischen Akademie der Wissenschaften. Er ist Mitglied im Institut für Dokumentologie und Editorik, in der Gesellschaft für Forschungssoftware - de-RSE und Co-Convener der AG „Research Software Engineering in den Digital Humanities“. Sein Forschungsinteresse liegt in der erfolgreichen Durchführung von Digital Humanities-Projekten und hier insbesondere in der nachhaltigen Softwareentwicklung, damit alle digitalen Forschungsergebnisse möglichst lange für die Nutzung erhalten bleiben. Kontakt: czmiel@bbaw.de

Ulrike Henny-Krahmer ist Juniorprofessorin für Digital Humanities an der Universität Rostock, Mitglied des Instituts für Dokumentologie und Editorik und Co-Convenerin der DHd-AG „Research Software Engineering in den Digital Humanities“. Ihre Forschung konzentriert sich auf Digitale Editionen und Textsammlungen, quantitative Textanalyse und die Evaluation und Nachhaltigkeit von digitalen Forschungsergebnissen. Kontakt: ulrike.henny-krahmer@uni-rostock.de

Daniel Jettka ist Wissenschaftlicher Mitarbeiter im Projekt NFDI4Culture, wo er im Arbeitsbereich „Forschungswerkzeuge und Datendienste“ tätig ist. Insbesondere arbeitet er an der Koordination der technischen Infrastruktur in NFDI4Culture mit und ist Teil der Beratungsagentur für nachhaltige Softwareentwicklung. Er ist Co-Convener der DHd-AG „Research Software Engineering in den Digital Humanities“. Kontakt: daniel.jettka@uni-paderborn.de

Bibliographie

Anzt, Hartwig, Felix Bach, Stephan Druskat, Frank Löffler, Axel Loewe, Bernhard Y. Renard, Gunnar Seemann, et al. 2020. “An environment for sustainable research software in Germany and beyond: current state, open challenges, and call for action [version 2; peer review: 2 approved].” *F1000Research* 2021, 9:295. https://doi.org/10.12688/f1000research.23224.2 .

Bastian, Mathieu, Sebastien Heymann und Mathieu Jacomy. 2009. “Gephi: an open source software for exploring and manipulating networks.” In *International AAAI Conference on Web and Social Media*. https://gephi.org/publications/gephi-bastian-feb09.pdf (zugegriffen: 26. Juli 2022).

Beck, Kent, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland und Dave Thomas. 2001. "Manifest für Agile Softwareentwicklung." http://agilemanifesto.org/iso/de/manifesto.html (zugegriffen: 27. Juli 2022).

Clean Code Developer. 2022. „Clean Code Developer.“ https://clean-code-developer.de/ (zugegriffen: 27. Juli 2022).

Eder, Maciej, Jan Rybicki und Mike Kestemont. 2016. “Stylometry with R: a package for computational text analysis.” *R Journal* 8(1): 107-21. https://journal.r-project.org/archive/2016/RJ-2016-007/index.html (zugegriffen: 26. Juli 2022).

Goble, Carole. 2014. “Better Software, Better Research.” Software Sustainability Institute. https://www.software.ac.uk/resources/publications/better-software-better-research (zugegriffen: 27. Juli 2022).

Halstenberg, Jürgen, Bernd Pfitzinger und Thomas Jestädt. 2020. *DevOps: Ein Überblick.* Wiesbaden: Springer.

Hettrick, Simon. 2020. „The growth and professionalisation of Research Software Engineering”. https://slides.com/simonhettrick/rse-professionalisation-cw20 (zugegriffen: 28. Juli 2022).

Katerbow, Matthias und Georg Feulner. 2018. „Handreichung zum Umgang mit Forschungssoftware“. Zenodo. https://doi.org/10.5281/zenodo.1172970 .

Martin, Robert C. 2009. *Clean Code. Refactoring, Patterns, Testen und Techniken für sauberen Code.* Frechen: mitp-Verlag.

Schrade, Torsten, Alexander Czmiel und Stephan Druskat. 2018. "Research Software Engineering und Digital Humanities. Reflexion, Kartierung, Organisation." In *DHd 2018. Book of Abstracts.* https://doi.org/10.5281/zenodo.4622564 .

SyncRO Soft SRL. 2022. „Oxygen XML Editor.“ https://www.oxygenxml.com/xml_editor.html (zugegriffen: 26. Juli 2022).