Bausteine Forschungsdatenmanagement
Empfehlungen und Erfahrungsberichte für die Praxis von Forschungsdatenmanagerinnen und -managern

Lösungsansätze zur automatisierten Erfassung von strukturierten Provenance Informationen in Forschungsdateninfrastrukturen am Beispiel von Analyse-Workflows in R

Arne Rümmler[i] Heiko Figgemeier[ii] Christin Henzen[iii]

2022

Zitiervorschlag

Rümmler, Arne, Heiko Figgemeier und Christin Henzen. 2022. Lösungsansätze zur automatisierten Erfassung von strukturierten Provenance Informationen in Forschungsdateninfrastrukturen am Beispiel von Analyse-Workflows in R. *Bausteine Forschungsdatenmanagement. Empfehlungen und Erfahrungsberichte für die Praxis von Forschungsdatenmanagerinnen und -managern* Nr. 1/2022: S. 85-102. DOI: 10.17192/bfdm.2022.1.8367.

[i]Technische Universität Dresden. ORCID: 0000-0001-8637-9071
[ii]Technische Universität Dresden. ORCID: 0000-0001-7802-5097
[iii]Technische Universität Dresden. ORCID: 0000-0002-5181-4368

Abstract

In den Umweltwissenschaften sind derzeit viele Forschungsprojekte datengetrieben und liefern Datensätze als wesentliche Ergebnisse. Das Publizieren von Daten nach den FAIR-Prinzipien stellt damit einen zentralen Faktor in der Entwicklung von Forschungsdateninfrastrukturen dar. Provenance-Informationen als Teil der Metadaten beschreiben die Herkunft der Daten und unterstützen damit die Verständlichkeit, Bewertung und Reproduzierbarkeit verfügbarer Forschungsdaten. Da eine manuelle und nachträgliche Erfassung strukturierter Provenance-Information zeitintensiv ist, besteht ein großer Bedarf an (teil-)automatisierten Lösungen, die ein nutzungsfreundliches und nahtloses Metadatenmanagement innerhalb einer Forschungsdateninfrastruktur ermöglichen.

Im Forschungsprojekt GeoKur, einem Projekt zur Kuration und Qualitätssicherung von Umweltdaten, erfolgt die Datenanalyse und -erzeugung in der Skriptsprache R. Dieser Erfahrungsbericht stellt daher die Evaluierung von Tools zur (teil-)automatisierten Erfassung von Provenance-Informationen in R-Skripten zusammen und beschreibt zwei selbst entwickelte Ansätze: (1) die Erzeugung von Provenance-Dateien mithilfe des selbst implementierten R-Pakets r2provo und (2) die direkte Publikation von Provenance-Informationen aus dem Analyseskript in ein Datenmanagementsystem mittels R-Paket ckanr.

1 Einleitung

In den letzten Jahren hat sich die Publikation wissenschaftlicher Daten als essenzieller Teil eines Forschungsergebnisses weiter durchgesetzt. Mit den FAIR-Prinzipien[1] wurden Guidelines und Metriken entwickelt, um die Auffindbarkeit (findability), die Zugänglichkeit (accessibility), die Interoperabilität (interoperability) und die Wiederverwendbarkeit (reusability) von Forschungsdaten zu verbessern. Darin wird Provenance – die Herkunft eines Datensatzes – als maßgebliche Information für die Bewertung der Wiederverwendbarkeit der Daten beschrieben. Provenance für Umweltforschungsdaten umfasst die durchgeführten Prozessierungsschritte, deren Eingangs(geo)daten und Ergebnisdaten sowie die beteiligten Akteur·innen[2]. Im strukturierten Provenance-Datenmodell PROV-DM[3] werden Datensätze als Entitäten, Prozessierungsschritte als Aktivitäten und Wissenschaftler·innen oder Institutionen als Agenten bezeichnet. Das

[1] Wilkinson, Mark D.; Dumontier, Michel; Aalbersberg, I. Jsbrand Jan; Appleton, Gabrielle; Axton, Myles; Baak, Arie et al. (2016): The FAIR Guiding Principles for scientific data management and stewardship. In: *Scientific data* 3:160018. DOI: https://doi.org/10.1038/sdata.2016.18.

[2] Moreau, Luc (2010): The Foundations for Provenance on the Web. In: *FNT in Web Science* 2 (2-3), 99–241. DOI: https://doi.org/10.1561/1800000010.

[3] PROV-DM, W3C Recommendation (2013): https://www.w3.org/TR/2013/REC-prov-dm-20130430/ (letzter Aufruf 17.12.2021).

PROV-DM beschreibt Provenance-Informationen als gerichteten Graph und ermöglicht damit deren maschinelle Verarbeitung.

Provenance-Informationen wurden vor wenigen Jahren überwiegend manuell erhoben. Inzwischen konnten für einige Anwendungsfälle Konzepte zur (teil-)automatisierten bzw. unterstützten Erfassung sowie strukturierte und austauschbare Provenance-Formate entwickelt werden. Die anwendungsfallspezifischen Individuallösungen und die generischen automatisierbaren Ansätze zusammenzuführen stellt immer noch eine Herausforderung dar. Der Aufwand ist in den verfügbaren Konzepten oft hoch und liegt entweder in der manuellen Erfassung der Provenance-Informationen oder in der Installation und Konfiguration benötigter Tools. Es existieren kaum Ansätze für die nahtlose Integration standardkonformer Provenance-Informationen in Forschungsdateninfrastrukturen, die das zusätzliche Erfassen prozessierungsspezifischer Parameter und das direkte Weiterverarbeiten unterstützen.

Provenance-Informationen können sich je nach Nutzungskontext und damit einhergehenden Informationsbedarfen u. a. in disziplinspezifischen Inhalten, Umfang, Granularität oder Formalisierungsgrad erheblich unterscheiden. Demzufolge verändern sich die Anforderungen an die Erfassung ebenfalls kontextspezifisch. In diesem Erfahrungsbericht wurden daher allgemeine und für einen konkreten Anwendungsfall aus den Umweltwissenschaften spezifische Kriterien zur Bewertung von Provenance-Erfassungs-Tools zusammengestellt und genutzt.

1.1 Anwendungsfall: Verarbeitung von Umweltdaten im Projekt GeoKur

Im Projekt GeoKur[4] werden Methoden, Tools und Best Practices zur Kuration und Qualitätssicherung von Geodaten über den gesamten Lebenszyklus der Forschungsdaten (Abbildung 1) entwickelt. Die zentrale Komponente der im Projekt aufgebauten Forschungsdateninfrastruktur ist ein gemeinsam genutztes Datenmanagementsystem (DMS), eine CKAN[5]-Instanz[6] mit einem vom Metadatenschema GeoDCAT[7] abgeleiteten Profil[8], welches insbesondere um Datenqualitätsaspekte und PROV-DM-konforme Provenance erweitert wurde.

In GeoKur erfolgt die Datenverarbeitung und -analyse in der Skriptsprache R. Das nachfolgende Skript (Listing 1) stellt einen Auszug eines komplexen Analyse-Workflows dar und dient als Beispiel für die Evaluierung der Tools und eigenen Ansätze.

[4] GeoKur Website: https://geokur.geo.tu-dresden.de/ (letzter Aufruf 17.12.2021).
[5] Open-Source CKAN: https://ckan.org/ (letzter Aufruf 17.12.2021).
[6] GeoKur-spezifisches DMS: https://geokur-dmp.geo.tu-dresden.de/ (letzter Aufruf 16.12.2021).
[7] GeoDCAT Schema: https://inspire.ec.europa.eu/good-practice/geodcat-ap (letzter Aufruf 17.12.2021).
[8] GeoKur Profil: https://doi.org/10.5281/zenodo.4916698.

Abbildung 1: Aspekte von GeoKur während des Forschungsdaten-Lebenszyklus‘

```
if (!require("raster")) install.packages("raster"); library ("raster")
if (!require("rgdal")) install.packages("rgdal"); library ("rgdal")

pollination <- raster("dataset1")
yieldRapeseed <- raster("dataset2")

# change crs
pollinationProj <- projectRaster(pollination, crs="+proj=longlat +datum=WGS84 +no_defs ")

# resample to 5 arcmin
pollinationRes <- resample(pollinationProj,yieldRapeseed)

# rearrange to table
outputTable <- cbind(as.data.frame(yieldRapeseed),as.data.frame(pollinationRes))

# remove 0 yields and NAs
outputTableFinal <- outputTable[which(
        outputTable$yieldRapeseed>0&!is.na(outputTable$pollination)),]

write.csv(outputTableFinal,"myOutputTable.csv")
```

Listing 1: Implementierungsbeispiel für die Prozessierung von Umweltdaten in R

Das Skript implementiert die Prozessierung von Umweltdaten zur Analyse von Rapserträgen zur Ermittlung von Bestäubungspotenzialen in Europa. Hierbei werden Daten aus zwei Rasterdatensätzen (Abbildung 2, oben: *Pollination Probability und Rapeseed Yield*) in einer gemeinsamen Tabelle kombiniert (Abbildung 2, unten: *Rapeseed Yield and Pollination*).

Pollination Probability wird dazu in ein anderes Koordinatenreferenzsystem projiziert (*Reproject*, Abbildung 2) und anschließend auf die Rasterzellengröße von *Rapeseed Yield* aggregiert (*Resample*, Abbildung 2). Dann erfolgt die Zusammenführung der Datensätze sowie das Entfernen fehlerhafter Werte (*Combine*, Abbildung 2).

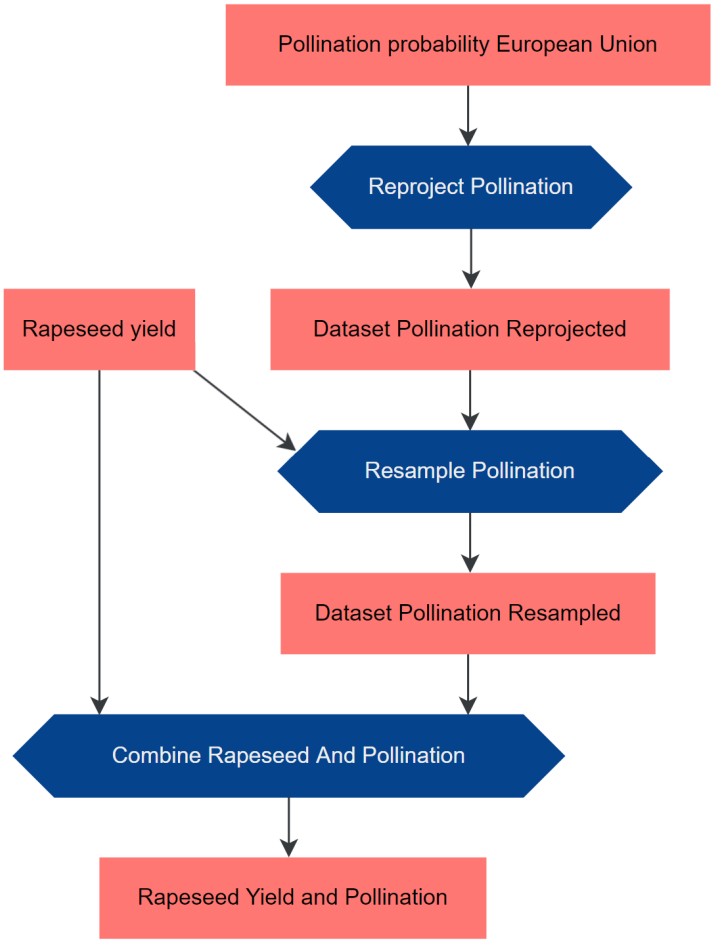

Abbildung 2: Provenance-Graph des Anwendungsbeispiels visualisiert im Webclient Prov-Viewer[9]; Datensätze visualisiert als rote Rechtecke, Verarbeitungsschritte in blau

[9] ProvViewer: https://geokur-dmp2.geo.tu-dresden.de/provViewer/ (letzter Aufruf 17.12.2021).

2 Evaluierung existierender Tools zur Erfassung von Provenance-Informationen

Die Transparenz wissenschaftlicher Workflows zur Erzeugung von Forschungsdaten ist ein wesentlicher Bestandteil der Bewertung der Nutzbarkeit der Daten für weitere Anwendungsfälle. Da der Quellcode oft komplex ist, werden Mechanismen zur Erfassung wesentlicher Bestandteile eines Workflows benötigt. Existierende Tools erfassen Provenance-Informationen **skriptbasiert** oder **annotationsbasiert**. Bei skriptbasierten Ansätzen führt das Aufrufen spezieller Funktionen aus dem Workflow-Skript zum (semi-)automatischen Aufzeichnen von Provenance-Informationen mit oder ohne spezifische Laufzeitparameter. Annotationsbasierte Ansätze erfordern das Einfügen qualifizierter Kommentare (Annotationen), um Provenance unabhängig von der Ausführung des Skripts zu erfassen.

2.1 Evaluationskriterien

Die im Rahmen des Projekts durchgeführte Evaluierung bestehender Provenance-Erfassungs-Tools erfolgte auf Basis zuvor festgelegter Ober- und Unterkriterien, die allgemeine Anforderungen an die Nutzbarkeit von Softwareprodukten, Anforderungen zur Erfassung von Provenance-Informationen im Forschungsdatenmanagement und spezifische Anforderungen des jeweiligen Anwendungsfalls umfassen (Tabelle 1). Für jedes Tool erfolgt eine Beschreibung und Diskussion positiv und negativ bewerteter Unterkriterien, sowie eine zusammenfassende Bewertung hinsichtlich der Oberkriterien (jeweils in eckigen Klammern und in Tabelle 2).

Installation

Nutzer·innen von Provenance-Erfassungs-Tools implementieren Workflows und sind damit Domänenexpert·innen mit heterogenen IT-Kenntnissen und teilweise eingeschränkten PC-Rechten. Daher muss das Tool im besten Fall ohne eine Installation auskommen bzw. mit geringem Aufwand zu installieren sein. Für eine nahtlose Integration in bestehende Arbeitsprozesse sollte das Tool in die existierende Arbeitsumgebung integriert werden können.

Konfigurierbarkeit

Die Konfigurierbarkeit beschreibt Optionen zur Festlegung der aufzuzeichnenden Informationen und zur Ergänzung zusätzlicher Informationen. Ein Erfassungs-Tool muss es Nutzer·innen ermöglichen, die aufzuzeichnenden Schritte des Workflows, den Detailgrad – bspw. ob (interne) Variablennamen und -belegungen erfasst werden oder

nicht – sowie ergänzende Informationen – wie z. B. von Funktionsnamen abweichende Anzeigenamen – festzulegen.

Nutzungsaufwand

Der Nutzungsaufwand umfasst verschiedene Aspekte, wie den Implementierungsaufwand, die Bedienbarkeit und damit einhergehende Erlernbarkeit sowie die Dokumentation des Tools. Der Implementierungsaufwand beschreibt die Menge des zu ergänzenden Codes zur Erfassung der Provenance-Informationen. Tools sollten möglichst geringe Ergänzungen erfordern, wobei diese per Definition in annotationsbasierten Ansätzen umfangreicher als in skriptbasierten Ansätzen sind. Außerdem sollte die Ergänzung in sinnvoll strukturierten Blöcken erfolgen, um die Lesbarkeit des Skripts wenig einzuschränken.

Interoperabilität

Die Interoperabilität der Erfassungstools wird durch die Nutzung standardisierter Datenmodelle und von der Community anerkannter bzw. offener Datenausgabeformate erreicht. Zur Beschreibung von Provenance-Informationen hat sich das domänenunabhängige PROV-DM etabliert, das sich in verschiedene gängige Formate, z. B. RDF/XML und Turtle, serialisieren und in verschiedenen Tools weiterverarbeiten lässt. In GeoKur muss die Ausgabe im PROV-DM oder darin konvertierbarer Serialisierung erfolgen.

Systemfeedback und Validierung

Das Systemfeedback und die Validierung von Erfassungs-Tools umfassen das Anzeigen von Warnhinweisen – z. B. Syntaxfehler – bzw. Problemen und die graphische Visualisierung der Ausgabe. Neben der formalen Validität der Ausgabe stellt die Schlüssigkeit des Provenance-Graphen ein Fehlerpotential dar. Unschlüssige Graphen können durch das Auslassen von Informationen oder fehlerhafte Verknüpfungen entstehen und geben den Inhalt des Skriptes falsch wieder. Werden solche Fehler nicht vom Tool selbst ausgeschlossen, muss die Überprüfung des resultierenden Graphen möglich sein, z. B. durch die Visualisierung des Graphen. Diese kann durch das Tool selbst oder durch interoperable Ausgabeformate unterstützt werden (→ Interoperabilität).

In GeoKur wurde eine Vielzahl an verschiedenen Tools zur Erfassung von Provenance-Informationen aus R-Skripten evaluiert[10]. Aufgrund der positiven Bewertung hinsichtlich der betrachteten Kriterien werden rdt/rdtLite und YesWorkflow näher vorgestellt.

[10] provDebugR, provExplainR, provTraceR, provAnalyzeR, provSummarizeR, provGraphR, recordr, NoWorkflow, RDataTracker.

Tabelle 1: Zusammenfassung der erarbeiteten Kriterien zur Evaluierung von Provenance-Erfassungs-Tools

Oberkriterien	**Unterkriterien**	
Einrichtung/Installation	Installation des Erfassungstools	*
	Einbindung in Arbeitsumgebung (z. B. in R)	‡
Konfigurierbarkeit	Inhalt (z. B. Prozessierungsschritte, Variablen und -belegung)	†, ‡
	Ergänzungen (z. B. Label für die graphische Visualisierung)	‡
Nutzungs-(aufwand)	Implementierungsaufwand	*
	Bedienbarkeit (u. a. Einfachheit, Erlernbarkeit, Selbstbeschreibungsfähigkeit)	*
	Dokumentation (z. B. Installation, Nutzung / Konfiguration)	*
Interoperabilität	Ausgabe als strukturierte(s) Datenmodell(e)	†, ‡
	Dateiformate der Ausgabe	†, ‡
Systemfeedback & Validierung	Warnung vor Syntaxfehlern	*
	Validitätsprüfung der Ausgabe bzgl. Datenmodell & -format	*
	Graphische Visualisierung zur Überprüfung der aufgezeichneten Informationen (z. B. auf Schlüssigkeit)	†,‡
* = Allgemeine Anforderung an Softwareprodukte † = Anforderung zur Erfassung von Provenance-Informationen im Forschungsdatenmanagement ‡ = Spezifische Anforderung im vorgestellten Anwendungsfall		

2.2 rdt/rdtLite

rdt bzw. *rdtLite*[11] sind skriptbasierte Extraktions-Tools, die Provenance-Informationen teil- oder vollautomatisiert während der Ausführung eines R-Skriptes erfassen bzw. benutzerdefinierte Konsoleneingaben aufzeichnen können. Sie generieren eine JSON-Datei im Datenmodell PROV-DM, bieten weiterhin Einstellungsmöglichkeiten zur Festlegung der Granularität der auszugebenden Provenance-Informationen an und stellen eine integrierte Funktion zur Visualisierung des Graphen bereit. Für das Beispielskript schlägt die Validierung der Datei mit dem Web-Tool ProvValidator und auch die Visualisierung fehl[12].

rdtLite wird als dokumentierte R-Bibliothek zur Einbindung in R bereitgestellt. Zur Laufzeit können u. a. die Ausführungszeit von Prozessen, temporäre Dateien oder verwendete R-Bibliotheken aufgezeichnet werden. Das Paket umfasst sowohl

[11] https://github.com/End-to-end-provenance/rdtLite (letzter Aufruf 17.12.2021).
[12] Validierungen beispielhafter Outputs mit dem ProvValidator (Teil der ProvToolbox) schlugen fehl. https://lucmoreau.github.io/ProvToolbox/ (letzter Aufruf 17.12.2021).

Funktionen zur Erfassung der Provenance-Informationen (1) von Eingaben in die R-Kommandozeile, als auch (2) eines kompletten R-Skripts mit dem Befehl *prov.run()* (Listing 2).

```
1 # run 'example_usecase.R' and gather provenance
2 prov.run("example_usecase.R")
```

Listing 2: Nutzung des Tools rdtLite

Obwohl die Inhalte des resultierenden Graphen anpassbar sind, sind die Einstellmöglichkeiten so gering, dass diese für den GeoKur-Anwendungsfall nicht ausreichen [Konfigurierbarkeit: negativ].

Aufgrund der fehlschlagenden Validierung des Outputs und der integrierten Visualisierung, werden die Ausgabe als strukturiertes Datenmodell, die Validierungs- sowie die Visualisierungsmöglichkeiten negativ bewertet. PROV-JSON ist das einzige Ausgabeformat [Interoperabilität und Systemfeedback & Validierung: negativ].

Die simple Bedienung, der hohe Automatisierungsgrad und der damit verbundene geringe Implementierungsaufwand werden positiv bewertet [Einrichtung/Installation und Nutzungsaufwand: positiv].

2.3 YesWorkflow

YesWorkflow[13] ist eine plattformunabhängige JAR-Bibliothek zur Extraktion und Visualisierung von Provenance-Informationen aus Skripten verschiedener Programmiersprachen, wie R oder Python, und ermöglicht eine annotationsbasierte Aufzeichnung Provenance-Informationen[14].

YesWorkflow interpretiert sprachenunabhängige schlüsselwortbasierte Annotationen in Kommentaren eines beliebigen Skripts und fasst diese zu einem Graphen zusammen. Die Erzeugung der Provenance-Informationen wird über einen konsolenbasierten Aufruf gestartet, eine Ausführung des Skriptes ist nicht notwendig[15]. Mögliche Fehler, z. B. falsch zugeordnete Variablennamen, sind aufgrund der graphischen Ausgabe einfach zu identifizieren.

[13] https://github.com/yesworkflow-org/yw-prototypes (letzter Aufruf 17.12.2021).

[14] McPhillips, T., Song, T., Kolisnik, T., Aulenbach, S., Belhajjame, K., Bocinsky, R. K., et al. (2015). YesWorkflow: A User-Oriented, Language-Independent Tool for Recovering Workflow Information from Scripts. *International Journal of Digital Curation*, 10, 298–313. https://doi.org/10.2218/ijdc.v10i1.370.

[15] Pimentel, João Felipe; Dey, Saumen; McPhillips, Timothy; Belhajjame, Khalid; Koop, David; Murta, Leonardo et al. (2016): Yin & Yang: Demonstrating Complementary Provenance from noWorkflow & YesWorkflow. In: Marta Mattoso und Boris Glavic (Hg.): Provenance and Annotation of Data and Processes, Bd. 9672. Cham: Springer International Publishing (Lecture Notes in Computer Science), 161–165.

Listing 3 zeigt das Beispielskript mit ergänzten Annotationen (grün geschrieben). Der Provenance-Graph und die entsprechende Visualisierung werden automatisch aus den Annotationen erzeugt (Abbildung 3).

```
# @BEGIN UseCase1
  # @IN input_pollination  @AS PollinationProbabilityEuropeanUnion @URI file:{db_pth}/...
  # @IN input_yieldRapeseed @AS RapeseedYield @URI file:{db_pth}/...
  # @OUT output @AS RapeseedYieldAndPollination @URI file: ...

  pollination  <- raster("dataset1")
  yieldRapeseed <- raster("dataset2")

  # @BEGIN ReprojectPollination
  # @IN input_pollination  @AS PollinationProbabilityEuropeanUnion @URI file:{db_pth}/ ...
  # @OUT pollination_proj  @AS DatasetPollinationReprojected
  pollinationProj  <- projectRaster(pollination , crs="EPSG") # change crs
  # @END ReprojectPollination

  # @BEGIN ResamplePollination
  # @IN pollination_proj  @AS DatasetPollinationReprojected @URI file:{db_pth}/...
  # @IN input_yieldRapeseed @AS RapeseedYield @URI file:{db_pth}/...
  # @OUT pollination_res @AS DatasetPollinationResampled
  pollinationRes  <- resample(pollinationProj,yieldRapeseed) # resample
  # @END ResamplePollination

  # @BEGIN CombineRapeseedAndPollination
  # @IN pollination_res  @AS DatasetPollinationResampled @URI file:{db_pth}/...
  # @IN input_yieldRapeseed @AS RapeseedYield @URI file:{db_pth}/...
  # @OUT output @AS RapeseedYieldAndPollination
  outputTable <- cbind(as.d.f.(yieldRapeseed),as.d.f.(pollinationRes))
  # @END CombineRapeseedAndPollination
# @END UseCase1
```

Listing 3: Skript des GeoKur-Anwendungsfalls mit YesWorkflow-Annotationen in grün

YesWorkflow ermöglicht eine Ausgabe in der Graph-Beschreibungssprache DOT[16] als gv-Datei[17], die mithilfe externer Software – z. B. GraphViz – visualisiert werden kann. Dabei können entweder die Datensätze oder Prozesse als Rechtecke dargestellt werden. Ein PROV-DM-konformer Export der Informationen und damit eine nahtlose Weiterverarbeitung in den GeoKur-spezifischen Anwendungen ist nicht möglich.

Das verwendete Datenmodell, nicht verfügbare Formate und die kommandozeilenbasierte Installation führen zu negativen Bewertungen in den entsprechenden Unterkriterien [Einrichtung/Installation und Interoperabilität: negativ].

Dem hohen Implementierungsaufwand des Tools steht eine einfache Bedienbarkeit gegenüber [Nutzungsaufwand: neutral].

[16] https://graphviz.org/doc/info/lang.html (letzter Aufruf 17.12.2021).
[17] https://graphviz.org/docs/outputs/canon/ (letzter Aufruf 17.12.2021).

Positiv wird die hohe Individualisierbarkeit des erzeugten Graphen in Bezug auf die erfassten Provenance- bzw. Zusatzinformationen bewertet. Auch die verfügbare graphische Darstellung und deren Anpassbarkeit wird positiv bewertet [Konfigurierbarkeit und Systemfeedback & Validierung: positiv].

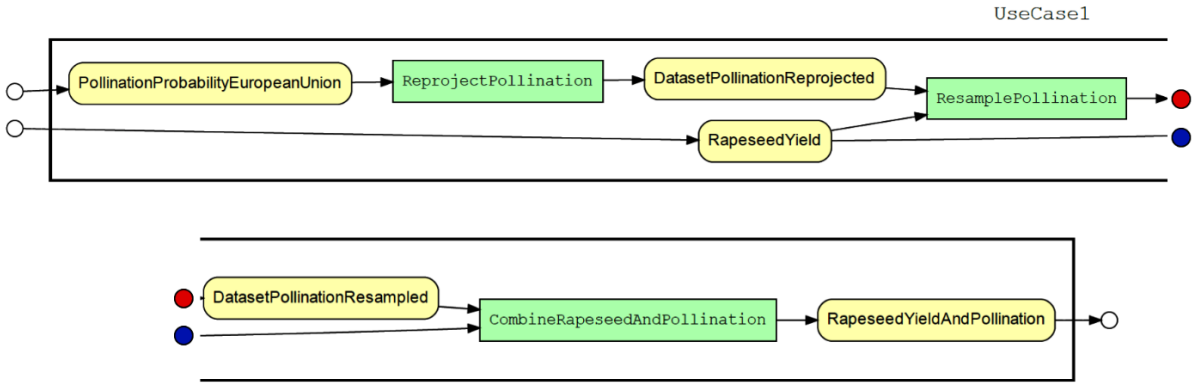

Abbildung 3: Mit YesWorkflow erzeugte Visualisierung des Provenance-Graphen für den Anwendungsfall (Listing 3); Prozesse als grüne Rechtecke, Daten als gelbe Container dargestellt

3 Projektspezifische Lösungsansätze und Implementierungen

In GeoKur sind vor allem die Kriterien Konfigurierbarkeit, Einrichtung/Installation und Interoperabilität von Bedeutung. Da keines der vorgestellten Tools in allen drei Kriterien positiv bewertet werden kann, wurden zwei skriptbasierte Ansätze entwickelt: (1) zur teilautomatisierten Generierung von PROV-DM-konformen Provenance-Informationen aus R-Skripten, prototypisch implementiert als R-Paket r2provo, und (2) basierend auf der im Projekt entwickelten Forschungsdateninfrastruktur und dem Open-Source-Paket ckanr[18], um Provenance-Informationen direkt aus einem R-Skript heraus im projektspezifischen DMS zu publizieren.

3.1 Skriptbasierter Ansatz mit dem Paket *r2provo*

r2provo[19] ist ein R-Paket, das Provenance-Informationen innerhalb eines R-Skripts als zusammenhängenden Provenance-Graphen entsprechend des PROV-DM erfasst und in RDF-Formaten ausgibt. Es kombiniert dabei die Vorteile der evaluierten Ansätze von

[18] https://github.com/ropensci/ckanr (letzter Aufruf 17.12.2021).
[19] https://github.com/GeoinformationSystems/R2ProvO (letzter Aufruf 17.12.2021).

rdtLite und YesWorkflow und ermöglicht eine teilautomatisierte Erfassung mit geringem Aufwand hinsichtlich Installation und Anpassung des Workflow-Skripts.

Wesentliches Ziel des Pakets ist es, Skriptautor·innen relevante Prozessierungsschritte für die Evaluierung und Visualisierung des Workflows selbst auswählen zu lassen. Dabei nutzt das Paket die typische Syntax eines Prozessierungsschrittes in Form eines Funktionsaufrufs mit mehreren Input-Variablen und einem Funktionsergebnis (Listing 4, Zeile 2). Ein Prozessierungsschritt kann dem Provenance-Graphen hinzugefügt werden, indem er in ein Tripel aus Funktionen eingebettet wird (Listing 4, Zeile 1 & 3):

```
1 eval(prov(quote(
2     Output <– method(Input_1, Input_2,  ...,  Input_n)
3 )))
```

Listing 4: Modifizierung eines R-Funktionsaufrufs zur Erfassung von Provenance-Information mittels r2provo

Die Funktionen *eval()* und *quote()* sind native R-Funktionen, die Funktion *prov()* wird von *r2provo* bereitgestellt und beinhaltet das Parsen, Verknüpfen und Konvertieren der Prozessierungsschritte in das PROV-DM. Die Funktion *quote()* wandelt das Argument, also den Prozessierungsschritt, in ein R-Objekt um, welches in der Funktion *prov()* ausgewertet und zum Provenance-Graphen hinzugefügt wird. Die Verknüpfung der Prozessierungsschritte erfolgt dabei über die Variablennamen – die Nutzung gleicher Variablennamen in verschiedenen Prozessierungsschritten führt zu deren Verknüpfung im Graphen. Die Funktion *eval()* führt den eingeschlossenen Befehl, hier den Prozessierungsschritt, anschließend in der R-Umgebung aus. *r2provo* kann auch für die Erfassung von Provenance-Informationen ohne Code-Ausführung genutzt werden, indem aus dem Funktions-Tripel der Funktionsaufruf *eval()* entfernt wird. Listing 5 zeigt, wie *r2provo* im Beispielskript angewandt wird – Ergänzungen zur Erfassung der Provenance-Informationen sind fett gekennzeichnet, z. B. wird die Projizierung des Rasters in ein anderes Koordinatensystem (Listing 5, Zeile 5) um das vorgestellte Funktions-Tripel (Listing 5, Zeile 4) ergänzt und damit dem Provenance-Graphen hinzugefügt. Die Schritte zur Konvertierung der Raster-Daten in R-Data-Frames sind zu fein granular und werden nicht mit aufgenommen (Listing 5, Zeile 14f).

Das Entfernen ungültiger Werte soll in den Provenance-Graphen übernommen werden. Da der entsprechende Befehl (Listing 5, Zeile 24) jedoch kein Funktionsaufruf ist, muss er in eine Funktion gekapselt werden, welche anschließend aufgerufen und in das Funktionstripel eingeschlossen wird (Listing 5, Zeile 22–29).

Das Paket r2provo beinhaltet keine automatische Validierung gegen das PROV-DM. Auch eine integrierte Visualisierung liegt nicht vor, jedoch kann die Schlüssigkeit des resultierenden Graphen, aufgrund der interoperablen PROV-DM-Ausgabeformate, mit den anfangs erwähnten Visualisierungs-Tools geprüft werden [Systemfeedback & Validierung: negativ].

```
1  init_prov()
2
3  # change crs
4  eval(prov(quote(
5      pollinationProj  <- projectRaster(pollination,  crs="+proj=longlat +datum=WGS84
           +no_defs ")
6  )))
7
8  # resample to 5 arcmin
9  eval(prov(quote(
10     pollinationRes <- resample(pollinationProj,yieldRapeseed)
11 )))
12
13 yieldRapeseed <- as.data.frame(yieldRapeseed)
14 pollinationRes <- as.data.frame(pollinationRes)
15
16 # rearrange to  table
17 eval(prov(quote(
18     outputTable <- cbind(yieldRapeseed, pollinationRes)
19 )))
20
21 # remove 0 yields  and NAs
22 cleanTable <- function(inTable) {
23     return(outputTable[which(inTable$yieldRapeseed > 0& !is.na(inTable$pollination)),  ])
24 }
25
26 eval(prov(quote(
27     finalTable  <- cleanTable(outputTable)
28 )))
29
30 write.csv(outputTableFinal,"myOutputTable.csv")
31 store_prov("use_case_short_prov.ttl")
```

Listing 5: Erfassung der Provenance-Information mit r2provo für das GeoKur-Beispielskript

Der Implementierungsaufwand ist gering, da der Quellcode des Workflows nicht verändert werden muss, sondern nur relevante Prozessierungsschritte in das vorgestellte Funktionstripel eingebettet werden müssen. Die Hauptaufgaben der Anwender·innen bestehen darin, die relevanten Prozessierungsschritte in Funktionen zusammenzufassen und die Verknüpfung der Prozessierungsschritte durch gleiche Variablennamen sicherzustellen. Hierbei werden jedoch Schwächen in der Bedienbarkeit identifiziert, da das Paket einen bestimmten Programmierstil forciert. Auch kann die Voranstellung des Funktionstripels Unübersichtlichkeit im Quellcode verursachen [Nutzungsaufwand: neutral].

Die Installation und Einbindung in R sind simpel. Der Inhalt des resultierenden Graphen kann von den Nutzer·innen durch die Auswahl relevanter Prozessierungsschritte gesteuert werden. Die Ausgabe des Graphen erfolgt im PROV-DM in einer beliebigen

RDF-Serialisierung [Konfigurierbarkeit, Einrichtung/Installation und Interoperabilität: positiv].

Das Paket wurde als Proof-of-Concept entwickelt und bietet diverse Entwicklungspotentiale, bspw. die Erweiterung der Funktion *prov()* um häufig auftretende Muster, die keinen Funktionsaufruf beinhalten oder um die Auswertung mehrerer Funktionsaufrufe innerhalb des Tripels. Dadurch kann der Implementierungsaufwand weiter reduziert und die Bedienbarkeit verbessert werden.

3.2 Forschungsdateninfrastrukturbasierter Ansatz mit dem Paket *ckanr*

Im GeoKur-Anwendungsfall ist die im Projekt entwickelte Forschungsdateninfrastruktur dazu geeignet, PROV-DM-konforme Provenance-Graphen zu erzeugen. Dazu werden im DMS für einen Datensatz die entsprechenden Inputs aus der Liste bereits eingetragener Datensätze ausgewählt (Abbildung 4).

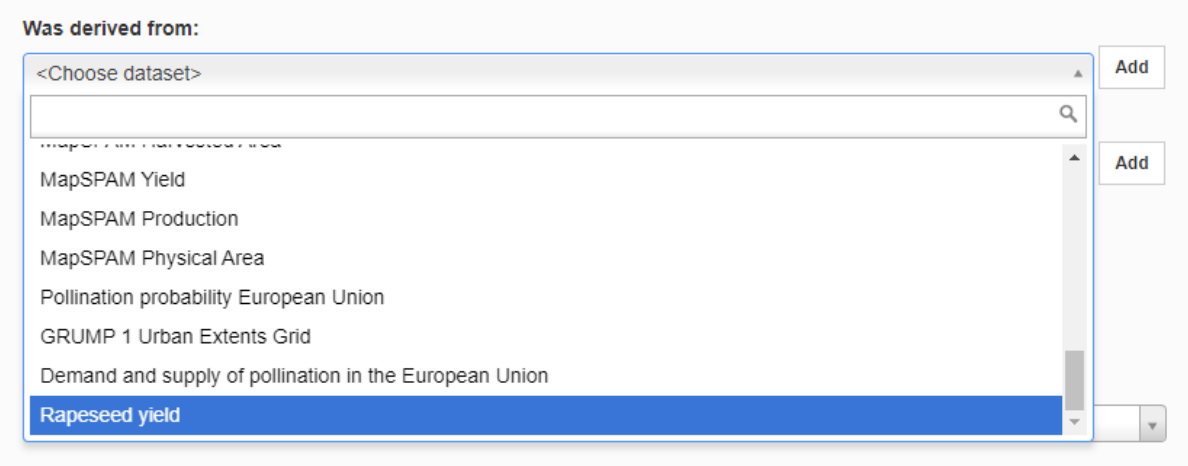

Abbildung 4: Auswahl der Input-Datensätze in der Katalogoberfläche zur Erstellung eines Provenance-Graphen

Alternativ zur Weboberfläche des DMS können die Metadaten über die CKAN-API[20] publiziert werden. Das R-Paket *ckanr* erleichtert die Verwendung der API aus einem R-Skript heraus und bietet Funktionen für das Erstellen, Aktualisieren und Löschen der Metadatensätze in einem CKAN.

Der hier entwickelte Ansatz basiert darauf, die für den Provenance-Graphen relevanten Metadaten mithilfe von *ckanr* zu erstellen bzw. zu aktualisieren und verbindet damit das Erfassen und Publizieren von Provenance-Informationen. Der Ansatz nutzt Aspekte der teilautomatisierten Erfassung von Provenance-Informationen. Wie auch im vorherigen Ansatz besteht ein wesentliches Ziel des Pakets darin, Skriptautor·innen die

[20] https://docs.ckan.org/en/2.9/api/ (letzter Aufruf 17.12.2021).

Entscheidung darüber zu überlassen, welche Prozessierungsschritte im resultierenden Provenance-Graphen enthalten sein sollen.

Listing 6 zeigt die Erstellung der Metadaten des Datensatzes *pollination_reprojected* aus dem Anwendungsbeispiel (Listing 1) und zugehörige Arbeitsschritte. Funktionen, die das Paket *ckanr* bereitstellt, sind fett gekennzeichnet. Die Erstellung der Metadaten erfolgt mithilfe der Funktion *package_create()*, welcher ein Vektor mit Metadatenelementen übergeben wird (Listing 6, Zeile 28-38). Da ein angepasstes Metadatenprofil und nicht das von CKAN bereitgestellte Schema implementiert ist, müssen die Metadaten als Liste mit der Bezeichnung *extras* übergeben werden (Listing 6, Zeile 29). Das GeoKur-Profil hat verpflichtende Metadatenfelder für Datensatzidentifikator, Datenproduzierende und Organisation (*name*, *contact_name* und *owner_org*). Die Merkmale Datenproduzierende und Organisation wurden als Variablen angelegt (Listing 6, Zeile 5f) und der *ckanr*-Funktion übergeben (Listing 6, Zeile 34f). Der Anzeigename des Datensatzes wird als *title* übergeben (Listing 6, Zeile 32). Die Provenance-Informationen werden schließlich über *was_derived_from*[21] durch Zuordnung der URL des Metadatensatzes des entsprechenden Inputs erfasst (Listing 6, Zeile 36), im Beispiel die CKAN-URL des Pollination-Datensatzes.

```
1  # configure CKAN connection
2  ckanr_setup("https://geokur-dmp.geo.tu-dresden.de/", key = "***")
3
4  # fill metadata variables
5  dataset_organization = «your_organization>"
6  script_author = «your_name>"
7  base_url <- "https://geokur-dmp.geo.tu-dresden.de/dataset/"
8
9  # get metadata set from CKAN
10 pollination_metadata <- package_show("demand-and-supply-of-pollination-in-the-european
       -union")
11
12 # review metadata
13 str(pollination_metadata)
14
15 # create url for metadata set
16 pollination_metadata_url <- paste0(dataset_base_url, pollination_metadata$id)
17
18 # get resource download url (target resource has index 1)
19 download_url_pollination <- pollination_metadata$resource[[1]]$url
20
21 # download dataset from CKAN and store as R Object called pollination
22 pollination <- raster(download_url_pollination)
23
24 # reproject the raster to WGS84
25 pollination_reprojected <- projectRaster(pollination, crs = "+proj=longlat +datum=WGS84
       +no_defs")
```

[21] In der GeoKur-CKAN-Instanz können auch Prozessmetadaten angelegt werden, wodurch die PROV-DM-Relationen *was_generated_by* und *used* ebenfalls zur Verfügung stehen.

```
26
27 # create new meta dataset in CKAN and store it as variable (dataset_reprojected_metadata).
28 pollination_reprojected_metadata <- package_create(
29   extras = c(
30     name = "dataset_pollination_reprojected",
31     title = "Dataset Pollination Reprojected",
32     conforms_to = "http://www.opengis.net/def/crs/OGC/1.3/CRS84",
33     owner_org = dataset_organization,
34     contact_name = script_author,
35     was_derived_from = pollination_metadata_url
36   )
37   )
```

Listing 6: GeoKur-Beispiel ergänzt um Code zur Erstellung eines neuen Metadatensatzes mittels *ckanr*

Zum Erfassen ist das Einfügen eines größeren Code-Blocks nötig, wobei Templates als Kopiervorlage vorbereitet werden können. Es liegt eine umfangreiche Dokumentation des Tools und zur Verwendung im Projektkontext[22] vor. Dennoch ist eine umfangreichere Einarbeitung als bei den anderen Tools erforderlich [Nutzungsaufwand: neutral].

Die Installation und Einbindung von ckanr in R ist einfach. Die Implementierung der Infrastruktur erhöht den initialen Aufwand zur Einrichtung der Umgebung, wird jedoch typischerweise durch ein IT-Team durchgeführt, wie z. B. im GeoKur-Projekt. Die Inhalte des aufgezeichneten Provenance-Graphen können konfiguriert und um zusätzliche Informationen ergänzt werden. Die Ausgabe des Graphen erfolgt im DMS gemäß dem PROV-DM in einer beliebigen RDF-Serialisierung. Die Validität des PROV-DM wird durch das DMS gewährleistet. Die Forschungsinfrastruktur in GeoKur-Projekt beinhaltet ebenfalls eine Provenance-Visualisierung. Durch die Verwendung der CKAN-API verhindert ckanr das Eintragen semantisch oder syntaktisch fehlerhafter Informationen in die für den Provenance-Graphen relevanten Felder [Einrichtung/Installation, Konfigurierbarkeit, Interoperabilität und Systemfeedback & Validierung: positiv].

4 Fazit

Es existieren viele Tools zur Erfassung von Provenance-Informationen aus Skripten mit unterschiedlichen funktionalen Schwerpunkten. In diesem Erfahrungsbericht wurden ergänzend zwei selbst entwickelte skriptbasierte Ansätze vorgestellt, die den Anforderungen des GeoKur-Anwendungsfalls genügen. Die Evaluierung der Tools erfolgte anhand festgelegter Kriterien (Tabelle 1 und Tabelle 2) mithilfe einer dreistufigen Skala von (+) für überwiegend positive Merkmale, (0) für positive und negative Merkmale, und (-) für überwiegend negative Merkmale.

[22] Anleitung: https://github.com/GeoinformationSystems/Guidance_Documents/tree/master/CKAN/api_from_r.

Die skript- bzw- annotationsbasierten Ansätze rdtLite und YesWorkflow nutzen unterschiedliche Methoden zur Erfassung von Provenance-Informationen. Obwohl rdtLite als Bibliothek in R integrierbar und nutzbar ist, ergibt sich durch die zu granulare Erfassung der Provenance-Informationen, die geringe Konfigurierbarkeit und derzeit fehlschlagende Validierung des resultierenden Graphen keine sinnvolle Anwendbarkeit im Projekt. Annotationen mittels YesWorkflow ermöglichen ein individuelles Konfigurieren und einfaches Nutzen des Tools. Die Installation ist vergleichsweise herausfordernd und die ausschließliche Bereitstellung der Provenance-Informationen als Visualisierung erlaubt keine interoperable Nutzung der Ergebnisse.

Tabelle 2: Übersicht der ausgewählten Tools und Bewertungen

	rdt/rdtLite[23]	YesWorkflow[24]	r2provo[25]	ckanr[26]
Datenmodell	PROV-DM (nicht valide)	DOT Language	PROV-DM	PROV-DM (CKAN-intern)
Einrichtung / Installation	+	-	+	+
Konfigurierbarkeit	-	+	+	+
Nutzungs-(aufwand)	+	0	0	0
Interoperabilität	(-)	-	+	+
Systemfeedback & Validierung	(-)	+	-	+

Da die genannten Tools für den GeoKur-Anwendungsfall (ohne Modifikation) keine Nutzung innerhalb der Forschungsdateninfrastruktur und basierend auf dem entwickelten Metadatenprofil ermöglichen, wurden zwei eigene Ansätze entwickelt, welche darauf abzielen, eine konfigurierbare und interoperable Erfassung von Provenance-Informationen zu ermöglichen. Die Tools *r2provo* und *ckanr* erzeugen interoperable PROV-DM-Outputs. *r2provo* kann direkt in R verwendet werden und erzeugt mit wenig Nutzungsaufwand einen konfigurierbaren Output. Obwohl *ckanr* einen höheren Nutzungsaufwand als *r2provo* aufweist, überzeugen die Integration in die Forschungsdateninfrastruktur, das hohe Maß an Konfigurierbarkeit sowie die Unterstützung bei effizientem Management und Visualisieren von Provenance-Informationen und anderen Metadaten.

[23] https://github.com/End-to-end-provenance/ (letzter Aufruf 17.12.2021).
[24] https://github.com/yesworkflow-org/yw-prototypes (letzter Aufruf 17.12.2021).
[25] https://github.com/GeoinformationSystems/R2ProvO (letzter Aufruf 17.12.2021).
[26] https://github.com/GeoinformationSystems/ckanr (letzter Aufruf 17.12.2021).

Provenance-Erfassungs-Tools bieten ein großes Potential für die Evaluierung wissenschaftlicher Datenprodukte, indem sie maßgebliche Informationen direkt aus den Analyse-Skripten (teil-)automatisiert erfassen, in Forschungsdateninfrastrukturen nahtlos weiterverarbeiten und damit den Aufwand für Datenproduzent·innen stark reduzieren.