Universität Rostock
Traditio et Innovatio

Max Tilman Kaseler, Ali Moghbelnaghsh,
Atefeh Pooladi, Chris Röhrs, Jasper Roloff

Einführung in das Thema

PROVENANCE

Sommersemester 2021

Gutachter
M. Sc. Tanja Auge

Studiengang
M. Sc. Informatik

Lehrstuhl
Lehrstuhl für Datenbank- und Informationssysteme

Abgabedatum
30. September 2021

Abstract

In this article Provenance is introduced with its various types and application domains. Four subtypes are distinguished. Meta Data Provenance is considered the most general one and includes all metadata of a given object. The Information System Provenance narrows to digital data created by information systems. When modelling workflows the Workflow Provenance is used. Insight into three dimensions will be given: Granularity, Domain and Form. To describe the origin of data within a database we utilize the Data Provenance. It can output the source data of a query result. Furthermore the W3C PROV standard is explained. Additionally a collection of tools, for instance Kepler, will be presented.

Zusammenfassung

In diesem Artikel wird Provenance in ihren verschiedenen Arten und Anwendungsbereichen vorgestellt. Es wird zwischen vier verschiedenen Provenancearten unterschieden. Die Meta Data Provenance gilt als die allgemeinste Provenance und beschreibt alle Metadaten von einem Objekt. Die Information System Provenance beschränkt sich auf die in einem Informationssystem entstandenen digitalen Daten. Wenn ein Arbeitsablauf modelliert werden soll, wird die Workflow Provenance verwendet. Bei dieser wird ein Einblick in die drei Dimensionen Granularität, Domäne und Form wird gegeben. Um die Herkunft von Daten innerhalb einer Datenbank zu beschreiben, wird die Data Provenance verwendet. Diese kann zu dem Ergebnis einer Anfrage die Quelldaten ausgeben. Weiterhin gibt es eine Einführung in den PROV-Standard des W3C. Außerdem werden verschiedene Tools, darunter Kepler, zum Umgang mit Provenance vorgestellt.

Keywords Provenance, FAIR, Data Provenance, Workflows, Metadata, Tools, W3C PROV

Inhaltsverzeichnis

1. Einführung

Die rasche Zunahme von Daten in verschiedenen Anwendungen, von Software über Forschungsdatenbanken bis hin zu Workflow-Systemen, macht es erforderlich, die Quelle der Daten zu kennen. Die Provenance bezieht sich nicht nur auf die zeitliche Vergangenheit eines Ergebnisses, sondern auch auf die Beziehungen zwischen dieser Ressource und anderen Einheiten, die bei der Erstellung des Endprodukts eine Rolle gespielt haben. Durch die Verwendung der Provenance kann jede Nutzerin und jeder Nutzer die Zuverlässigkeit der Ressource bewerten, da sie/er weiß, wer die Daten im Laufe ihrer Geschichte beeinflusst hat und wer zu jenen beigetragen hat. Außerdem ist die Reproduzierbarkeit ein wichtiges Prinzip in der wissenschaftlichen Forschung, das durch die Verwendung der Provenance ermöglicht wird. Die Reproduzierbarkeit ermöglicht es den Forschern, ihre Ergebnisse zu überprüfen, und bietet außerdem die Möglichkeit, neue Methoden mit anderen Methoden zu vergleichen, die zuvor in reproduzierbaren Publikationen vorgestellt wurden.

Als leicht verständliche Einführung in das Themenfeld Provenance soll dieser Literaturbericht dienen. Er zeigt auf, wo sich die Provenance verbirgt, auch wenn nicht direkt von ihr gesprochen wird. Ein gemeinsames Beispiel aus einer fiktiven Universität führt durch die einzelnen Kapitel und erklärt theoretische Konzepte. Zusätzlich wird Software vorgestellt, welche sich explizit auf die Provenance konzentrieren.

Aufbau dieses Dokuments Dieses Dokument ist in elf Kapitel eingeteilt. Zunächst wird in Kapitel 2 das themenübergreifende Studierendenbeispiel eingeführt. Im Kapitel 3 wird eine Übersicht über die vier verschiedenen Provenancearten gegeben und darunter im Abschnitt 3.2 werden die *FAIR*-Prinzipien vorgestellt. In welchen Bereichen Provenance Anwendung findet, wird im Kapitel 4 dargelegt. Die Metadata Provenance wird als erste der Provenancearten in Kapitel 5 erklärt. Direkt im Anschluss blicken wir auf Kapitel 6 mit der Information System Provenance. Im darauf folgenden Kapitel 7 wird die Workflow Provenance vorgestellt. Kapitel 8 präsentiert die Data Provenance mit den Provenance-Halbringen 8.2 als Schwerpunkt. Verschiedene Provenance-Tools folgen im 9. Kapitel, mit Hauptaugenmerk auf Kepler. Das Kapitel 10 beschäftigt sich mit der Provenance im Web und geht schwerpunktmäßig in den *PROV*-Standard ein. Schließlich fasst Kapitel 11 die Ergebnisse der Arbeit zusammen.

2. Allgemeines Beispiel

In diesem Artikel wird ein einfaches Beispiel einer Universitätsdatenbank verwendet, die die Prüfungsanmeldung auswertet. In diesem fiktiven Beispiel werden die Unterrichtsangebote verschiedener Professoren erwähnt, damit sich die Studierende zur Prüfung der gewünschten Lehrveranstaltung anmelden können. Der Aufbau der Datenbank und ihrer Tabellen sowie die an diesen Tabellen durchgeführten Anfragen werden im Folgenden detailliert beschrieben.

2.1. Aufbau der Datenbank

Die Universitätsdatenbank besteht aus fünf Tabellen, die jeweils einen Primärschlüssel und zugehörige Attribute enthalten, die in jeder der folgenden Tabelle ausführlich beschrieben werden. Die erste Tabelle ist die Tabelle der Studierenden. Sie enthält vier Attribute mit den folgenden Titeln: Matrikelnummer, Vor- und Nachname und Studiengang der Studierenden. Die Matrikelnummer ist als Primärschlüssel dieser Tabelle dargestellt. Wie in Tabelle A.1 gezeigt, werden die Labels S_1 bis S_8 verwendet, um die Tupel zu identifizieren, von denen jedes die Eingabedaten jedes Studierenden repräsentiert. In dieser Tabelle sind die Informationen von acht Studierenden zu sehen. Die zweite Tabelle ist die Veranstaltungstabelle. Sie enthält zwei Attribute mit den folgenden Titeln: Kursnummer und Titel, wobei die Kursnummer der Primärschlüssel dieser Tabelle ist. Wie in Tabelle A.2 gezeigt, wurden die Labels C_1 bis C_9 verwendet, um die Tupel zu identifizieren, von denen jedes die Eingabedaten für jede Veranstaltung darstellt. In der Veranstaltungstabelle sind die Informationen von neun Veranstaltungen sichtbar. Als dritte Tabelle ist die Tabelle der Dozenten in Tabelle A.3 zu sehen. In dieser Tabelle werden zwei Attribute mit Titeln angezeigt: Kursnummer und Name der Professoren. Die Kursnummer spielt die Rolle des Fremdschlüssels und enthält Informationen über die Veranstaltungen, die von jedem Professor angeboten werden. In der Dozententabelle sind die Informationen von neun Professoren ersichtlich. Tabelle A.4 zeigt die Teilnehmertabelle, in der zwei Attribute angezeigt werden: die Kursnummer und die Matrikelnummer. Beide sind Fremdschlüssel der Kurstabelle sowie der Studierendentabelle und listen auf, an welcher Veranstaltung jeder Studierende teilgenommen hat. Die Teilnehmertabelle enthält 29 Eingabedaten. Die letzte Tabelle A.5 ist die Notentabelle, die die vier Attribute Lehrveranstaltungsnummer, Matrikelnummer, Semester und Note enthält, von denen sowohl die Lehrveranstaltungsnummer als auch die Matrikelnummer Fremdschlüssel sind. Diese Tabelle zeigt, in welchem Semester jeder Studierende welche Lehrveranstaltung mit welcher Note erhalten hat.

2.2. Anfrage und Anfrageergebnis

Im Folgenden werden vier Anfragen bezogen auf die Hochschuldatenbank am Studierendenbeispiel untersucht und bei jeder der erhaltenen Anfrageantworten wird auch die Datenherkunft (***how***, ***why***, ***where*** und ***lineage***) untersucht.

Die Anfrage 2.1 ruft ohne Wiederholung alle Studiengänge aus der Studierendentabelle ab, in denen Studierende eingeschrieben sind. Tabelle 2.1 zeigt das Anfrageergebnis, in der alle drei Fragen zur Datenherkunft und ***lineage*** untersucht werden.

```
SELECT DISTINCT study_course FROM students;
```

Anfrage 2.1: Anfrage 1

study_course	*how*	*why*	*where*	*lineage*
Computer Science	$S_4 \oplus S_5 \oplus S_6$	$\{\{S_4\},\{S_5\},\{S_6\}\}$	STUDENTS	S_4, S_5, S_6
Computer Science for Teaching	S_1	$\{\{S_1\}\}$	STUDENTS	S_1
Electrical Engineering	$S_3 \oplus S_7$	$\{\{S_3\},\{S_7\}\}$	STUDENTS	S_3, S_7
Mathematics	S_2	$\{\{S_2\}\}$	STUDENTS	S_2
Theoretical Computer Science	S_8	$\{\{S_8\}\}$	STUDENTS	S_8

Tabelle 2.1: Ergebnis der ersten Anfrage mit Provenance

Die Anfrage 2.2 zeigt die Liste der Studierenden, die an der Veranstaltung mit der Kursnummer „005“ teilgenommen haben. In dieser Anfrage werden die Tabellen von Studierenden, Kurs und Teilnehmer zusammengefügt. Das Ergebnis dieser Anfrage ist in Tabelle 2.2 zu sehen.

```
SELECT
    s.s_id, s.firstname, p.p_id, p.course_nr
FROM
    students s INNER JOIN
    participants p ON s.student_id = p.student_id
WHERE
    p.course_nr = '005'
;
```

Anfrage 2.2: Anfrage 2

s_id	firstname	p_id	course_nr	*how*	*why*	*where*	*lineage*
S_4	Elisabeth	P_{16}	005	$S_4 \otimes P_{16}$	$\{\{S_4, P_{16}\}\}$	STUDENTS, PARTICIPANTS	S_4, P_{16}
S_7	Jack	P_{17}	005	$S_7 \otimes P_{17}$	$\{\{S_7, P_{17}\}\}$	STUDENTS, PARTICIPANTS	S_7, P_{17}

Tabelle 2.2: Ergebnis der zweiten Anfrage mit Provenance

Die Anfrage 2.3 zeigt, welche Studierende von welchen Dozenten unterrichtet werden. In dieser Anfrage werden drei Tabellen miteinander verknüpft. In der Antwort auf diese Anfrage ist also kein Tupel aus der Teilnehmertabelle, daher ist sie im Abschnitt ***where*** nicht beteiligt. Allerdings sind die beiden Tabellen von Studierenden und Dozenten durch diese Tabelle verbunden, und diese Tabelle ist wirksam in der Anfrageantwort. Also werden Tupel der Teilnehmer in den

Abschnitten ***how***, ***why*** und ***lineage*** angezeigt, siehe dazu Tabelle 2.3.

```
SELECT s.firstname, l.fullname
FROM students s, participants p, lecturers l
WHERE s.student_id = p.student_id
AND p.course_nr = l.course_nr;
```

Anfrage 2.3: Anfrage 3

fstname	**fullname**	***how***	***why***	***where***	***lineage***
Donald	Lec. A	$S_1 \otimes P_1 \otimes L_{1.2}$	$\{\{S_1, P_1, L_{1.2}\}\}$	STUDENTS, LECTURERS	$S_1, P_1, L_{1.2}$
Donald	Prof. A	$S_1 \otimes P_1 \otimes L_{1.1}$	$\{\{S_1, P_1, L_{1.1}\}\}$	STUDENTS, LECTURERS	$S_1, P_1, L_{1.1}$
Sarah	Lec. A	$S_2 \otimes P_2 \otimes L_{1.2}$	$\{\{S_2, P_2, L_{1.2}\}\}$	STUDENTS, LECTURERS	$S_2, P_2, L_{1.2}$
Sarah	Prof. A	$S_2 \otimes P_2 \otimes L_{1.1}$	$\{\{S_2, P_2, L_{1.1}\}\}$	STUDENTS, LECTURERS	$S_2, P_2, L_{1.1}$
Elisabeth	Lec. A	$S_4 \otimes P_3 \otimes L_{1.2}$	$\{\{S_4, P_3, L_{1.2}\}\}$	STUDENTS, LECTURERS	$S_4, P_3, L_{1.2}$
Elisabeth	Prof. A	$S_4 \otimes P_3 \otimes L_{1.1}$	$\{\{S_4, P_3, L_{1.1}\}\}$	STUDENTS, LECTURERS	$S_4, P_3, L_{1.1}$
John	Lec. A	$S_5 \otimes P_4 \otimes L_{1.2}$	$\{\{S_5, P_4, L_{1.2}\}\}$	STUDENTS, LECTURERS	$S_5, P_4, L_{1.2}$
John	Prof. A	$S_5 \otimes P_4 \otimes L_{1.1}$	$\{\{S_5, P_4, L_{1.1}\}\}$	STUDENTS, LECTURERS	$S_5, P_4, L_{1.1}$
⋮	⋮	⋮	⋮	⋮	⋮

Tabelle 2.3: Ergebnis der dritten Anfrage (Ausschnitt) mit Provenance

Die Anfrage 2.4 zeigt die Noten und die Anzahl der Häufigkeit jeder dieser Noten für die Lehrveranstaltung mit der Lehrveranstaltungsnummer „002“. Das Anfrageergebnis sowie die Untersuchung ihrer Datenquelle sind in Tabelle 2.4 zu sehen.

```
SELECT grade, count(*) AS gcount
FROM grades
WHERE course_nr = '002'
GROUP BY grade
ORDER BY grade;
```

Anfrage 2.4: Anfrage 4

grade	**gcount**	***how***	***why***	***where***	***lineage***
1.0	1	G_8	$\{\{G_8\}\}$	GRADES	G_8
1.3	2	$G_6 \oplus G_9$	$\{\{G_6, G_9\}\}$	GRADES	G_6, G_9
2.0	1	G_{10}	$\{\{G_{10}\}\}$	GRADES	G_{10}
2.3	1	G_7	$\{\{G_7\}\}$	GRADES	G_7
3.3	1	G_{11}	$\{\{G_{11}\}\}$	GRADES	G_{11}
3.7	1	G_5	$\{\{G_5\}\}$	GRADES	G_5

Tabelle 2.4: Ergebnis der vierten Anfrage mit Provenance

3. Allgemeiner Überblick über Provenance

Was ist Provenance? Mit der weiten Verbreitung von Daten ist die Notwendigkeit der Herkunft spürbar geworden. Da Daten in einer Vielzahl von Anwendungen, von wissenschaftlichen Workflow-Systemen bis hin zu Textableitungen, sozialen Medien und mehr, schnell wachsen, hat das Wissen, woher die Daten stammen und wie sie produziert wurden, viele interessiert. Provenance oder Quelle eines Endprodukts hat die Möglichkeit, zahlreiche Informationen zu erfassen: jede Person, jeder Gegenstand, jede Zeit, jeder Ort und jede Handlung, die zur Herstellung eines Endprodukts beigetragen hat [LP15].

Im Allgemeinen ist Provenance als Metadaten zu betrachten, die anstatt der Datenbeschreibung den Prozess der Datenproduktion beschreiben. Aus verschiedenen Gründen ist es sehr wichtig, die Daten zu erfassen und zu verarbeiten. Diese Gründe sind die Qualität, die Gewährleistung der Reproduzierbarkeit und die Schaffung von Vertrauen im Endprodukt [HDB17].

Provenance wird sowohl auf die Zeitgeschichte des Datenprodukts bezogen, als auch auf die Verbindung zwischen Datenprodukt und den an der Erstellung der Informationen beteiligten Elementen. Mit anderen Worten, Provenance gibt eine Antwort auf folgende Fragen: Warum, wie, wo, wann und von wem sind die Informationen generiert worden?

Warum brauchen wir Provenance? Aus der oben erwähnten Definition der Provenance geht hervor, weswegen Provenance verwendet wird: aus den Gründen des Vertrauens, der Validierung und der Reproduzierbarkeit eines Endprodukts. Im Folgenden wird ein Überblick gegeben, wie wichtig die Reproduzierbarkeit eines Endprodukts ist. Diese Notwendigkeit, Experimente zu reproduzieren, mit dem Ziel ihrer Validierung und Erweiterung, bestand schon immer. Dass man vergangene Ergebnisse nutzt und überprüft, ist ein Standardmuster in allen wissenschaftlichen Bereichen. Die Reproduzierbarkeit gibt den Gutachtern die Möglichkeit, die Ergebnisse zu testen und ermöglicht auch Vergleiche zwischen neuen Methoden und den bereits in reproduzierbaren Publikationen vorgestellten Methoden. Schließlich führt die Reproduzierbarkeit des Endprodukts zur erheblichen Verbesserung der Wirkung, der Vision und der Qualität der Forschung [FC18].

3.1. Provenance-Hierarchie

Es gibt verschiedene Arten von Provenance, die nach den Bedürfnissen der Nutzung differenziert werden. Die vier Haupttypen der Provenance werden in der folgenden Abbildung 3.1 hierarchisch vom allgemeinsten Zustand auf der untersten Hierarchieebene bis zum spezifischsten Typ auf der obersten Hierarchieebene, dargstellt. Vier Arten von Provenance werden entsprechend ihrer allgemeinen und spezifischen Verwendung benannt: Metadaten-Provenance, Informationssystem-Provenance, Workflow-Provenance bzw. Data-Provenance. Die unterste Ebene der Hierarchie enthält den größtmöglichen Zustand zum Entwerfen von Data-Provenance sowie eine größere Anzahl von Prozessen, und wenn wir in dieser Hierarchie aufsteigen, nehmen beide Fälle ab, wie in Tabelle 3.1 dargestellt. Im Folgenden wird jeder Typ im Detail untersucht [HH16].

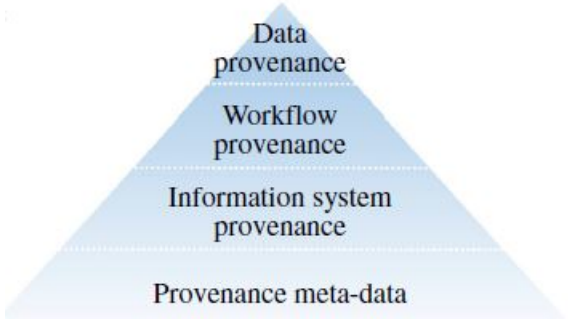

Abbildung 3.1: Provenance-Hierarchie [HH16]

Prov. type	Process type	Prov. data model
data provenance	structured query language	based on specific data model and query language of corresponding process type
workflow prov.	workflows	standards-based
information system provenance	processes supported by information systems	standards-based
prov. meta-data	anything	anything

Tabelle 3.1: Restrictions on process type and provenance data model on provenance types. [HH16]

3.1.1. Metadata Provenance

Diese Art der Provenance, die als die allgemeinste Form der Provenance bekannt ist und die größtmöglichen Prozesse und Modellierungen von Provenance-Informationen umfasst, bezieht sich auf jede Art von Informationen, die die Quelle und den Prozess der Erstellung eines Endprodukts erklären. Durch die Verwendung dieser Art wird dem Nutzer zahlreiche Möglichkeiten zur Modellierung, Speicherung und Zugänglichkeit der Provenance verschiedener Arten von Endergebnissen und ihres Erstellungszyklus angeboten. Metadaten und Metadaten-Provenance beinhalten zwei verschiedene Argumente. Metadaten sind auf Informationen über Daten bezogen, während die Metadaten-Provenance Informationen über den Datenextraktionsprozess angibt [HH16].

3.1.2. Information System Provenance

Auf der zweiten Hierarchieebene gibt es die Informationssystemherkunft, bei der die Art des Prozesses auf den Prozess innerhalb des Informationssystems beschränkt ist. Die Provenance des Informationssystems umfasst die Provenance für verschiedene Arten von Produktionsprozessen innerhalb einer eigenen Standarddarstellung der Provenance entsprechenden Informationssystems. Von denen ist eine der wichtigsten der W3C-Standard [MBC13], bei der es sich um eine spezifische Definition von Datenmodellen, Ontologien und Komponenten für den Zugriff und die Anfrage der Provenance handelt [HH16].

3.1.3. Workflow Provenance

Auf einer höheren Ebene als die Informationssystem-Provenance in der Hierarchie befindet sich die Workflow-Provenance, auf der der Produktionsprozess eingeschränkter wird. Die auf Workflow basierenden Systeme sind als Option im Gegensatz zu improvisierten Methoden zur Entwicklung rechnerisch-logischer Tests entstanden [HDB17; DF08]. Forscher werden bei der Konzeption und Bewältigung des Untersuchungszyklus von Workflow-Systemen unterstützt, damit sie die Analyseaufgaben mithilfe der Verwaltung der in jeder Phase verbrauchten und produzierten Daten erstellen und wiederverwenden können. Darüber hinaus zeichnet es die Datenquelle für die zukünftige Verwendung auf [DF08]. Der Arbeitsablauf kann als einen aus Knoten und Kanten bestehenden Graphen vorgestellt werden, wobei Knoten oder die Module in der Lage sind, den Betrieb insgesamt mit einigen Eingaben, Ausgaben und Parametern darzustellen. Und von Kanten wird ein vordefinierter Datenfluss oder Kontrollstrom zwischen diesen Modulen gezeigt. Die anhand des Workflows erhaltenen Informationen berücksichtigen verschiedene Strukturen und Granularitäten der Provenance in verschiedenen Anwendungsbereichen, in denen Attribute des Arbeitsprozessdiagramms verschoben werden [HDB17]. Die Workflow-Provenance ist eine Information über die Workflow-Eigenschaften, die Verbesserung dieser Eigenschaften und die Implementierung des Workflows [KMF18].

3.1.4. Data Provenance

Die Data Provenance steht auf der höchsten Ebene der Hierarchie, von der die spezifischste Verarbeitungsart und das Datenmodell der Datenherkunft umgefasst wird. Diese Hierarchieebene basiert auf einem bestimmten Datenmodell und einer Anfragesprache, die in einer relationalen Datenbank verarbeitet werden [HH16]. In der Data Provenance werden Verbindungen zwischen einem Endprodukt und der Informationsquelle untersucht. Die Datenherkunft wird in einer relationalen Datenbank verwendet, sodass jedes Tupel sehr genau verfolgt und analysiert werden kann. Es gibt drei entscheidende Fragen, die mithilfe von Data Provenance beantwortet werden können:

***Where*?** Woher kommt ein Ergebnis einer Anfrage? Das bedeutet, welche Tupel beteiligt waren, um das Ergebnis einer Anfrage zu erhalten. Es gibt unterschiedliche Granularitäten in dieser Datenherkunft, wenn verschiedene Datenbanken beteiligt waren. Aus welcher spezifischen Quelldatenbank stammen die einzelnen Ergebnisse? Wenn verschiedene Relationen beteiligt sind, kommt das Ergebnis einer Anfrage aus einer oder mehreren dieser Relationen, oder man könnte sagen, aus jeder Relation werden bestimmte Tupel gewählt. Zum Beispiel, im Studierendenbeispiel hat Sarah Morgan die Absicht, sich zur Prüfung „Operating Systems“ anzumelden, siehe Anfrage 3.1. Das Ergebnis der Anfrage (Tabelle 3.2) besteht dabei aus bestimmten Tupeln, welche die Studierenden-ID, die Kursnummer (Primärschlüssel in der Studierendentabelle und der Kurstabelle) sowie die Teilnehmer-ID enthalten.

Why*?** Warum wurde diese Anfrageantwort erzeugt? Die ***why-Provenance ist ähnlich zur ***where***-Provenance. Das bedeutet, warum diese Anfrageantwort erhalten wurde und welche Tupel

benötigt wurden, um diese Anfrageantwort zu erhalten. Hier werden die minimalen Zeugen berücksichtigt.

How? Wie wurde diese Anfrageantwort erzeugt? Dies bedeutet, wie die Daten durch die Anfrage manipuliert wurden, um diese Anfrageantwort erhalten [CCT09].

Lineage Durch ***lineage*** wird die Beziehung zwischen den Tupeln des Endprodukts und den anfänglichen Tupeln umgefasst [CCT09].

Die Lineage der Ausgabetupel (Sarah Morgan, Operating Systems) basiert auf Anfrage 3.1. Sie stammen aus den Tabellen Students (S_4), Course (C_2) und Participants (P_8), wobei die Students (S_4), die Course (C_2) und die Teilnehmertabelle (P_8) die Substanzen von Students-, Course- und Participants bezeichnen. Enthalten sind die Tupel S_4, C_2 und P_8. Diese drei Tupel sind der einzige Zeuge von Quelltupeln für unsere Ausgabe. Und keine anderen Quelltupel sind Teil des Zeugen. Der Zeuge ist eine Sammlung von Datenbankeingabewerten, um zu bestätigen, dass als Ergebnis der Anfrage Tupel des Endprodukts entwickelt werden [CCT09].

```
SELECT
    Students.First_Name + Last_Name AS FullName, s_id,
    c_id AS Course_nr, Title AS Course, Participants_id
FROM
    Students, Participants, Courses
WHERE
    Students.Student_id = Participants.Student_id AND
    Courses.Course_id = Participants.course_nr AND
    First_Name = 'Elisabeth' AND Title = 'Operating System'
;
```

Anfrage 3.1: Anfrage für Anmeldung zur Prüfung

FullName	S_id	Cours	Course	P_id	how	why	where	lineage
Elisabeth Harrison	S_4	C_2	Operating System	P_8	S_4*C_2*P_8	{{S_4, C_2, P_8}}	Students, Courses	S_2, C_2, P_6

Tabelle 3.2: Ergebnis von Anfrage für Anmeldung zur Prüfung

3.2. FAIR-Prinzipien

Datenmanagement ist ein wichtiger Ansatz im Bereich der wissenschaftlichen Forschung. Die ordnungsgemäße Archivierung von Daten, insbesondere von wichtigen Daten, die eine langfristige Pflege erfordern, sowie die Notwendigkeit der Wiederverwendung bereits veröffentlichter Daten im Interesse des wissenschaftlichen Fortschritts sind einige der Gründe für die Notwendigkeit der Verwaltung wissenschaftlicher Daten, um diese Daten in zukünftigen Studien wiederverwenden und auswerten zu können [Wil+16].

Um die Beschaffung wichtiger Daten und die Zusammenarbeit bei Forschungsprojekten zu erleichtern, wurden die FAIR-Prinzipien aufgestellt, die in dem Artikel [Wil+16] beschrieben werden. Zu diesen Grundsätzen gehören die folgenden:

- **Findability**: bedeutet, dass Daten aus allen verfügbaren Projekten auffindbar sind und in einer Datenquelle gespeichert werden, wobei alle Daten durch einen Identifikator gekennzeichnet sind.
- **Accessability**: Der zweite genannte Grundsatz ist die „Zugänglichkeit“, die das Auffinden aller Daten anhand eines Identifikators beschreibt.
- **Interoperability**: Der dritte Grundsatz ist die „Interoperabilität“, die besagt, dass die Daten eine gemeinsame Sprache verwenden müssen, um veröffentlicht werden zu können, und dass sie sich leicht auf andere Daten beziehen können.
- **Reusability**: Der vierte Grundsatz ist die „Wiederverwendbarkeit“, die besagt, dass die Daten verwandte Eigenschaften haben müssen, dass sie auf bestimmte Daten zugreifen können und dass sie eine bestimmte Quelle haben müssen.

Eine Übersicht über Aspekte dieser vier Grundsätze ist in Abbildung 3.2 hinterlegt.

Findability	*Accessability*
F1. (Meta)daten enthalten einen weltweit eindeutigen und dauerhaften Identifikator. F2. die Daten mit umfangreichen Metadaten beschrieben werden. F3. die Metadaten enthalten eindeutig und explizit den Identifikator der Daten, die sie beschreiben. F4. (Meta-)Daten sind in einer durchsuchbaren Ressource registriert oder indiziert.	A1. (Meta-)Daten sind anhand ihrer Kennung über ein standardisiertes Kommunikationsprotokoll abrufbar. A1.1 das Protokoll ist offen, frei und universell implementierbar. A1.2 das Protokoll ermöglicht ein Authentifizierungs- und Autorisierungsverfahren, falls erforderlich. A2. die Metadaten sind zugänglich, auch wenn die Daten nicht mehr verfügbar sind.
Interoperability	*Reusability*
I1. (Meta-)Daten verwenden eine formale, zugängliche, gemeinsame und breit anwendbare Sprache zur Wissensdarstellung. I2. (Meta-)Daten verwenden Vokabulare, die den FAIR-Grundsätzen folgen. I3. (Meta)daten enthalten qualifizierte Verweise auf andere (Meta)daten.	R1. Meta(daten) sind mit einer Vielzahl von genauen und relevanten Attributen reichhaltig beschrieben. R1.1. (Meta)daten werden mit einer klaren und zugänglichen Datennutzungslizenz freigegeben. R1.2. (Meta-)Daten sind mit einer detaillierten Provenienz verbunden. R1.3. (Meta-)Daten erfüllen domänenrelevante Gemeinschaftsstandards.

Abbildung 3.2: Zusammenfassung der FAIR-Prinzipien

4. Anwendungen von Provenance

Durch die Entstehung großer Datenmengen, wurde die Kenntnis der Provenance ein sehr wichtiger Faktor für die Entwicklung neuer Anwendungen. Provenance umfasst die verwendeten primären Informationsquellen sowie die Entitäten und Prozesse, die sich an der Erstellung des Endprodukts beteiligen. Anhand der Provenance ist auf die Verlässlichkeit der betreffenden Daten zu schließen und es kann auch nachvollzogen werden, wie die verschiedenen Informationsquellen eingebunden sind [Gro+12].

Artikel [HDB17] enthält eine Klassifizierung von Anwendungen der Provenance basierend auf drei Hauptideen:

1. Verständlichkeit
2. Reproduzierbarkeit
3. Qualität und Gültigkeit eines Prozesses

Wie in der Abbildung 4.1 unten gezeigt, unterscheidet sich jede der Anwendungen basierend auf der Zielgruppe und den Benutzern. Die Benutzertyperkennung bereitet einem die Möglichkeit, in Zukunft maßgeschneiderte und optimierte Anwendungen zu verschiedenen Formen von Verbrauchern und Herstellern von Provenance zu entwickeln [HDB17].

Im Allgemeinen ist die Zielgruppe auf jeden Benutzer zu beziehen, der die Provenance aufzeichnet oder verwendet. *Expert* bezieht sich auf Entitäten, die den Prozess verwalten oder ausführen, für den die Provenance gesammelt wurde. *All* zeigt Experten und Endbenutzer für ein Endprodukt. Ein Endbenutzer steht für eine Person, die nicht wissen muss, wie ein Endprodukt erstellt wurde. Und für *self* wird Provenance-Produzent und Provenance-Nutzer an denselben Experten eingeschränkt [HDB17].

In diesem Abschnitt werden Anwendungen erläutert, für die die Lösungen von Provenance erstellt wurden. Es sollte beachtet werden, dass die vorgeschlagenen Lösungen für eine Anwendung auch für andere Anwendungen funktionieren können. Der Grund könnten dieselben allgemeinen Herausforderungen im Hintergrund dieser Programme sein [HDB17].

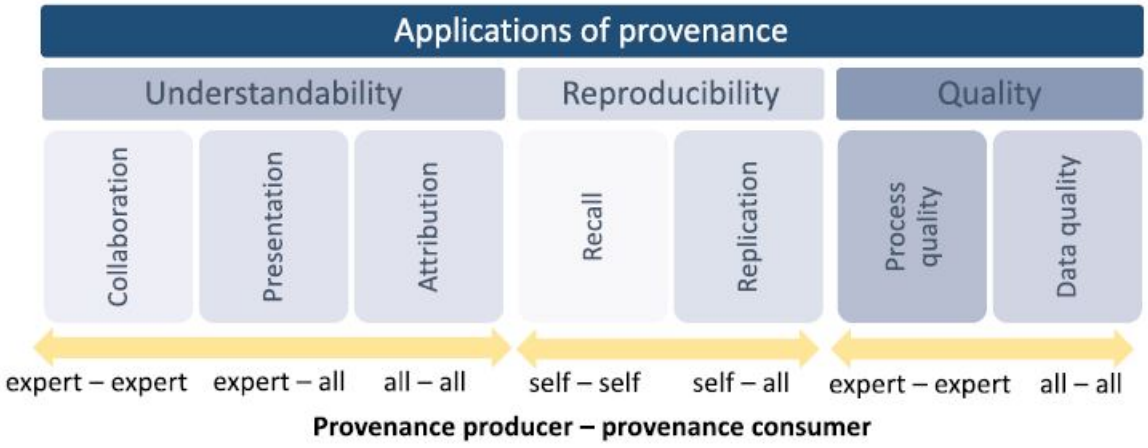

Abbildung 4.1: Classification application of provenance [HDB17]

Beispielsweise die Neugestaltung mittlerer Verarbeitungszustände unterstützt einen Spezialisten in der Überprüfung, warum verschiedene Vorbereitungsschritte zur Erstellung eines Objektes erforderlich waren. Auch ermöglicht es Forschern, die Folgen voneinander präzise zu wiederholen. Kenntnis von Bearbeitungszwischenzuständen hat den weiteren Ablauf eines Prozesses zur Folge [HDB17].

Die Zuordnung von Informationen zu Quellen stellt eine weitere Herausforderung dar. Dies führt offensichtlich dazu, Veränderungen und neue Formen der Interaktion den zusammenarbeitenden Designern zuzuschreiben und eine deutlichere und sicherere Darstellung bestimmter Realitäten zu ermöglichen [HDB17].

Expert und Endnutzer am Studierendenbeispiel Anhand des Studierendenbeispiels sind wir in der Lage, die Rolle des Experten und des Endnutzers besser zu verstehen. Nehmen wir an, ein Student hat die Absicht, sich zu einer Prüfung anzumelden. Um das zu machen, muss er seinen Antrag an das Prüfungsamt schicken. Vor der Anmeldung der gewünschten Veranstaltung zur Prüfung sind im Prüfungsamt einige Voraussetzungen zu prüfen, z. B. ob die erforderlichen Punkte für Hausaufgaben von dem Studenten erreicht wurden und ob die Veranstaltung, die er belegen will, in seiner Studienordnung steht oder nicht.

Endbenutzer: Der Endnutzer ist einer, der nicht wissen muss, wie ein Prozess funktioniert. Im obigen Beispiel lässt sich der Student als Endnutzer betrachten.

Expert: Ein Experte ist jemand, der weiß, wie ein Prozess funktioniert, um das Endergebnis zu erreichen. Im obigen Beispiel kann das Prüfungsamt als Experte betrachtet werden. Das Prüfungsamt muss ermitteln, ob der Student die erforderliche Hausarbeitsnote für seine Veranstaltung erreicht hat. Voraussetzung dafür ist, dass diese Veranstaltung in seiner Studienordnung enthalten ist.

4.1. Verständlichkeit

Mit Verständlichkeit werden die Ergebnisse oder Prozesse beschrieben, die für das Publikum deutlicher werden müssen. Sie informiert darüber, welche Art von Informationen übertragen werden und wie diese Übertragung erfolgt. In diesem Zusammenhang bietet es Anwendern und Fachpublikum Informationen zur Vorbereitung des Endprodukts. Verständlichkeit enthält *Collaboration*, *Presentation* und *Attribution*. Jeder von ihnen wird im Folgenden genauer untersucht [HDB17].

Collaboration Durch die Arbeit mit mehreren Benutzern an einem Projekt wird Zusammenarbeit geschafft. Es ist wichtig, dass man die Rollen und Aktionen der Beteiligten kennt. Provenance ist in der Lage, relevante Informationen zu übermitteln und den Gruppenmitgliedern zu ermöglichen, die Handlungen der anderen besser zu verstehen. Es ist klar, dass sich dieser Abschnitt auf die Expertengruppe konzentriert, in der jedes Mitglied Provenance erstellt oder verwendet [Rag+16].

Präsentation Bezüglich der Verständlichkeit ist die Art der Präsentation als sehr wichtig anzusehen. Präsentation steht in direktem Zusammenhang mit proportionaler Visualisierung sowie Dateninteraktion. Um ihre Verständlichkeit zu verbessern, braucht man unbedingt Kenntnisse darüber, wie Informationen gewonnen werden. Präsentation und Visualisierung sind miteinander verbunden, damit man die Informationsquelle leicht beobachten, führen und verfolgen kann. Node-Link-Diagramme werden verwendet, um die Informationsquelle zu visualisieren. „Diese Art der Visualisierung zeigt, wie die Dinge miteinander verbunden sind, indem Knoten / Kanten und Verbindungslinien verwendet werden, um ihre Verbindungen darzustellen und die Art der Beziehungen zwischen einer Gruppe von Einheiten zu verdeutlichen“ [Dat]. Aber die Verwendung dieser Art von Methode für sehr große Quelldatensätze verwirrt einen. Daher wird das *Sankey-Diagramm* für umfangreiche Daten verwendet. Dieses Diagramm gruppiert die Knoten des Diagramms der Knoten-Link-Herkunft und betont die Datenmenge, die zwischen den Aktivitäten ausgetauscht wird, um das Diagramm mit vielen Daten übersichtlich zu halten. Es werden auch andere Diagramme verwendet, zum Beispiel das *Radial-Diagramm*, um die Beziehungen zwischen und die Führung durch unterschiedliche Auflösung der Quelle zu zeigen. Eine weitere Darstellung ist das *Lamport-Diagramm*, um die verfolgten Ursprünge in verteilten Systemen zu zeigen. Bei dieser Anwendung von Provenance liegt die Aufnahme in der Verantwortung von Experten, und Verbraucher sind in der Regel Nicht-Experten, für die die entsprechende Visualisierung und ausreichende Interaktionen geschaffen werden sollen [HDB17].

Attribution In diesem Abschnitt beschränkt sich die Verständlichkeit auf einen Aspekt des Produktionsprozesses, wer oder was jeden Teil des Produktionsprozesses verfasst hat. Diese Informationen sind sehr wichtig und nützlich, um das Urheberrecht und die Eigentumsrechte an den Daten zu ermitteln, ihre Zitierung zu ermöglichen oder die Haftung für fehlerhafte Daten zu bestimmen. Die Visualisierung fällt in den Bereich der Kommunikation zwischen den Nutzern, da sie alle von jedem Mitarbeiter ausgeführten Aktionen klar zeigt [HDB17]. Visualisierung existiert in verschiedenen Systemen, die Provenance unterstützen. Beispielsweise wird in *Provenance-Aware Storage Systems* (kurz PASS) auch Visualisierung berücksichtigt [Mun$^+$06]. Ein PASS ist ein Aufbewahrungssystem, das die Aufgabe hat, das Original, die Installation und die Historie eines Produkts automatisch zu bewahren und zu sammeln. „PASS unterstützt die Zuordnung auf der Ebene von Befehlszeilenausführungen, wobei die Provenance Benutzer- und Gruppeninformationen für bestimmte Befehlszeilenausführungen aufzeichnet“ [Mun$^+$06] [HDB17].

Cloud-Infrastruktur In der Cloud-Infrastruktur existieren große Herausforderungen für eine effektive Datenverfolgung. Diese Herausforderungen haben zwei Hauptgründe:

1. Fehlendes Datenverfolgungswerkzeug für integrierte Clouds
2. Konzipierung der Aufzeichnungsmechanismen nur aus einer systemorientierten Perspektive

Man bedarf einiger datenzentrierten Protokollierungstechniken, sodass sich die Datenaktivität

auf allen Servern verfolgen lässt [Sue+13].

Die bisher genannten Systeme erfassen und stellen Metainformationen zur Verfügung, die explizit zugänglich sind. Es ist aber außerdem möglich, dass manchmal die Zuschreibung extrahiert werden muss, weil sie implizit in der Information verborgen ist. Zu den Beispielen gehört die Rekonstruktion der Herkunft von Informationen durch geheime Bedingungen zwischen sozialen Informationen. Dies beinhaltet das Erkennen von Kunden, die die Nachricht eines anderen Kunden anpassen, ohne dass sie ausdrücklich darauf hinweisen [HDB17; Nie+15].

4.2. Reproduzierbarkeit

Quelleninformationen werden von Forschern benötigt, damit sie Techniken und Methoden replizieren können, die die gleichen Ergebnisse erzielen. Ein wichtiges Merkmal der Datenreproduzierbarkeit ist das Verständnis sowie die Hervorhebung der Art und Weise der Datenextraktion durch Benutzer, damit sie diesen Daten vertrauen können. Es lässt sich sagen, dass Reproduzierbarkeit eine wichtige Rolle für die Wissenschaft spielt, denn die Reproduktion eines Forschungsergebnisses in einer wissenschaftlichen Methode ist ein Prinzip. Aber auch in anderen Bereichen ist es sehr wichtig [Mor11].

Reproduzierbarkeit bezieht sich auf verschiedene Begriffe. Im Folgenden werden drei Begriffe aus diesem Bereich erläutert [ACM20]:

Repeatability: Die Schätzung kann von einer ähnlichen Gruppe, die eine ähnliche Schätzungsstrategie, einen ähnlichen Schätzungsrahmen, unter ähnlichen Arbeitsbedingungen, in einem ähnlichen Gebiet und mit verschiedenen Vorbedingungen verwendet, mit ausdrücklicher Genauigkeit erhalten werden. Für computergestützte Tests bedeutet dies, dass ein Analytiker seine eigenen Berechnungen zuverlässig wiederholen kann.

Reproducibility: Die Schätzung kann von einer anderen Gruppe, die eine ähnliche Schätzungsstrategie, einen ähnlichen Schätzungsrahmen, unter ähnlichen Arbeitsbedingungen, im gleichen oder einem anderen Gebiet auf zahlreichen Vorläufen verwendet, mit der gleichen Genauigkeit durchgeführt werden. Für Berechnungsversuche bedeutet dies, dass eine unabhängige Gruppe ein ähnliches Ergebnis unter Verwendung der eigenen Artefakte des Urhebers erzielen kann.

Replicability: Die Schätzung kann mit ausdrücklicher Genauigkeit von einer anderen Gruppe, einem anderen Schätzungsrahmen, in einem anderen Bereich auf verschiedenen Experimenten erhalten werden. Für computergestützte Untersuchungen bedeutet dies, dass eine autonome Gruppe ein ähnliches Ergebnis erzielen kann, indem sie Artefakte verwendet, die sie völlig unabhängig erweitert.

Die Reproduzierbarkeit ist in zwei Unterkategorien unterteilt, abhängig davon, welche Institution die Quelle produziert oder konsumiert [HDB17].

Recall Die Erinnerung (Recall) an die einzelnen Schritte des Prozesses ist sehr wichtig, damit man ein Ergebnis erhalten kann, um es zu reproduzieren. Durch Aufzeichnen der Quelle und Verarbeitung unterstützt diese Recall Benutzer beim Sammeln von Informationen. Der Recall erzielt die für die Verarbeitung verantwortlichen Stellen, wobei der Erzeuger und der Verbrau-

cher der Quelle gleichermaßen in Betracht gezogen werden. Die Quelle für den Recall kann beinhalten, welche Zwischenschritte der Verarbeitung Änderungen verursacht haben, wie Parameter eingerichtet wurden und wie sie geändert wurden. Erinnerungen sind wichtig, damit man sich an Dinge erinnern kann, die im Prozess bereits getan wurden, und um die Entwicklung des Prozesses zu beschleunigen. Sie sind auch bei langen und intermittierenden Analysen und sogar in kurzen Intervallen wichtig, da die Benutzer während des Prozesses nicht dazu fähig sind, sich genau an ihre Analyse zu erinnern [HDB17]. Recall in der Provenance wird meist als Bestandteil anderer Anwendungen (Präsentation, Replikation usw.) verwendet, sodass nachvollziehbar ist, welche analytischen Aufgaben zuvor durchgeführt wurden, welche Ergebnisse sich bewährt haben und welche Aufgaben in Zukunft erledigt werden sollten [Rag+16].

Replikation Ein weiterer Grund für die Verwendung von Herkunftsinformationen ist die Möglichkeit, eine frühere Analyse zu reproduzieren. Die Replikation ist eine Anwendung von Recall, um ein Ergebnis nach der Analyse zu wiederholen und zu bestätigen. Infolgedessen kann die analytische Replikation dazu verwendet werden, überarbeitete Analysen zu erweitern, wenn die Parameter geändert wurden. Durch die Replikation werden bereits durchgeführte Analysen validiert und fortgeführt und die Wahrscheinlichkeiten können während der Analyse vollständig identifiziert werden, was sehr wichtig ist [Rag+16]. Bei der Replikation ist das Zielpublikum nicht nur der Autor des Endergebnisses, sondern alle Forscher, die auf diesem Gebiet gearbeitet haben. Dafür müssen die erfassten Quellen so reichhaltig sein, dass dieses Ergebnis von anderen Experten reproduziert werden kann [HDB17].

4.3. Qualität

Mit der Provenance lässt sich die Qualität eines Endergebnisses oder die Qualität eines Produktionsprozesses überprüfen. Die Prozessüberprüfung wird normalerweise von Experten durchgeführt. Die Datenqualität kann jedoch auch vom Endnutzer überprüft werden. Der Endnutzer ist jemand, der nicht wissen muss, wie die Daten gewonnen wurden. Die Qualität wird in zwei Bereiche unterteilt, Prozessqualität und Datenqualität [HDB17].

Prozessqualität Prozessqualität ist eine Methode, anhand deren die Bewertung und der Nachweis von Qualität ermöglicht wird. Qualität kann aus verschiedenen Aspekten wie Korrektheit, Leistung und Fehlertoleranz bestehen. Die Provenance wird im Zusammenhang mit der Prozessqualität zur Überwachung und Fehlersuche in Programmen verwendet, damit verschiedene Aspekte der Qualität zu bewerten sind. Die Prozesse sind in sehr unterschiedlicher Weise zu betrachten. Bei Provenance kann es sich um den Zustand von Webdiensten handeln, um Entwicklern bei der Auswahl des richtigen Dienstes für eine bestimmte Anwendung zu helfen. Oder sie können verwendet werden, um eine Verbesserung von Eingabewörterbüchern beim Data Mining sowie bessere Kennzeichnung von Entitäten in unstrukturierten Aufgaben oder Quellensätzen für verteilte Systeme und den Schutz vor Sicherheitsproblemen wie böswilligen und datenpolitischen Fehlern zu ermöglichen [HDB17].

Datenqualität Hier sind verschiedene Dimensionen und Metriken zur Bewertung der Datenqualität umfasst, einschließlich Vollständigkeit, Genauigkeit, Aktualität oder Vertrauen. Zielgruppe sind hier alle Nutzer [HDB17].

5. Metadata Provenance

Metadata Provenance wird als Informationen über die Erstellung, Zuordnung und Kopie des Datensatzes von verwalteten Daten bezeichnet. Von Provenance-Metadaten wird die Verbindung zwischen zwei Varianten von Datenobjekten gezeigt und sie werden immer dann erzeugt, wenn eine andere Form eines Datensatzes erstellt wird. Zum Beispiel sind hier der Name des Programms zu erwähnen, jenes das neue Formular erstellt hat, die Commit-ID des Programms, das in einem Code-Versionskontrollsystem verwendet wird, die Kennungen einiger anderer Datensätze oder Datenobjekte, die möglicherweise bei der Erstellung der neuen Version verwendet wurden [CAS].

Von den Provenance-Metadaten werden die Transformation der Eingabedaten in die Ausgabedaten detailliert beschrieben. So erhöhen sie die Reproduzierbarkeit der Berechnung massiv [BHK18].
Zusammenfassend lässt sich sagen, dass es sich bei Metadaten und Metadaten-Provenance um zwei unterschiedliche Argumente handelt: Metadaten beziehen sich auf Informationen über Daten, während Metadaten-Provenance Informationen über den Datenextraktionsprozess sind. Metadaten Provenance bezieht sich auf jede Art von Informationen, die die Quelle und den Prozess der Erstellung eines Endprodukts erklären. Durch die Verwendung dieses Typs hat der Benutzer zahlreiche Möglichkeiten zur Modellierung, Speicherung und zum Zugriff auf die Provenance verschiedener Produkttypen und deren Erstellungsprozess.

Metadata-Provenance im Studierendenbeispiel Wir wollen die Metadaten-Provenance anhand des folgenden Beispiels in der Universitätsdatenbank Untersuchen. Angenommen, der Name „Elisabeth“ ist in der Universitätsdatenbank als Student registriert. Dieses registrierte Tupel hat einige Metadaten, die in der Abbildung 5.1 aufgeführt sind. Nur einige der aufgeführten Metadaten in der Abbildung 5.1 können für die Datenvalidierung und Quellenauthentifizierung verwendet werden. Sie werden als Metadaten-Provenance bezeichnet. Zum Beispiel Erstellungsdatum, Ersteller, Tabellenname. Die genannten Metadaten stammen aus diesem registrierten Tupel, das zur Identifizierung der Datenquelle erforderlich ist.
Andere Metadaten wie der Spaltentyp oder die maximal definierte Zeichengröße in der Abbildung 5.1 werden nicht zur Identifizierung der Quelle verwendet und sind nur Metadaten.
Es ist zu beachten, dass die Provenance immer mit Metadaten angezeigt wird, aber nicht alle Metadaten müssen notwendigerweise Provenance sein.

	ColumnName	ColumnType	TableName	CharectarMaxLength	creator	CreateDate
Elisabeth Harrison	FullName	nchar	Student	50	Admin	6/30/2021 17:22

Abbildung 5.1: Metadata Provenance im Studierendenbeispiel

6. Information System Provenance

In diesem Abschnitt wird Provenance von Informationssystemen definiert, sodass diese Art von Provenance auf Prozesse beschränkt ist, die digitale Informationen innerhalb von Informationssystemen generieren, z. B. Big Data Processing Pipelines [HDB17].

Information System Provenance wird als Provenance von verschiedenen Arten von Produktionsprozessen innerhalb des Informationssystems betrachtet, sodass sie durch Informationssysteme unterstützt werden, die einer Standarddarstellung von Provenance entsprechen [HH16]. Zusammenfassend wird die Information System Provenance als Metadaten definiert, die für einen Informationsverbreitungsprozess gesammelt werden und die auf der Grundlage des Inputs, des Outputs und der Parameter des Prozesses berechnet werden können. [HDB17].

Information System Provenance im Studierendenbeispiel Nehmen wir an, dass StudIP, LSF und Prüfungsportal jeweils ein Informationssystem sind und eine eigene Datenbank haben. Die beiden Informationssysteme LSF und Prüfungsportal sind mit StudIP verbunden. Hier nehmen wir zum Beispiel StudIP als das gesuchte Informationssystem (wie in der Abbildung 6.1 dargestellt). Bei der Information System Provenance spielt es keine Rolle, bei wem und wie die Daten aus andere Informationssystem in das gesuchte Informationssystem gekommen sind. Denn diese Aufgabe ist der Workflow-Provenance. Hier sind die Metadaten, die den Prozess der Veröffentlichung von Informationen zeigen, z. B. Metadaten für das Herunterladen von Folien und Hochladen und so weiter als Information System Provenance betrachtet werden.

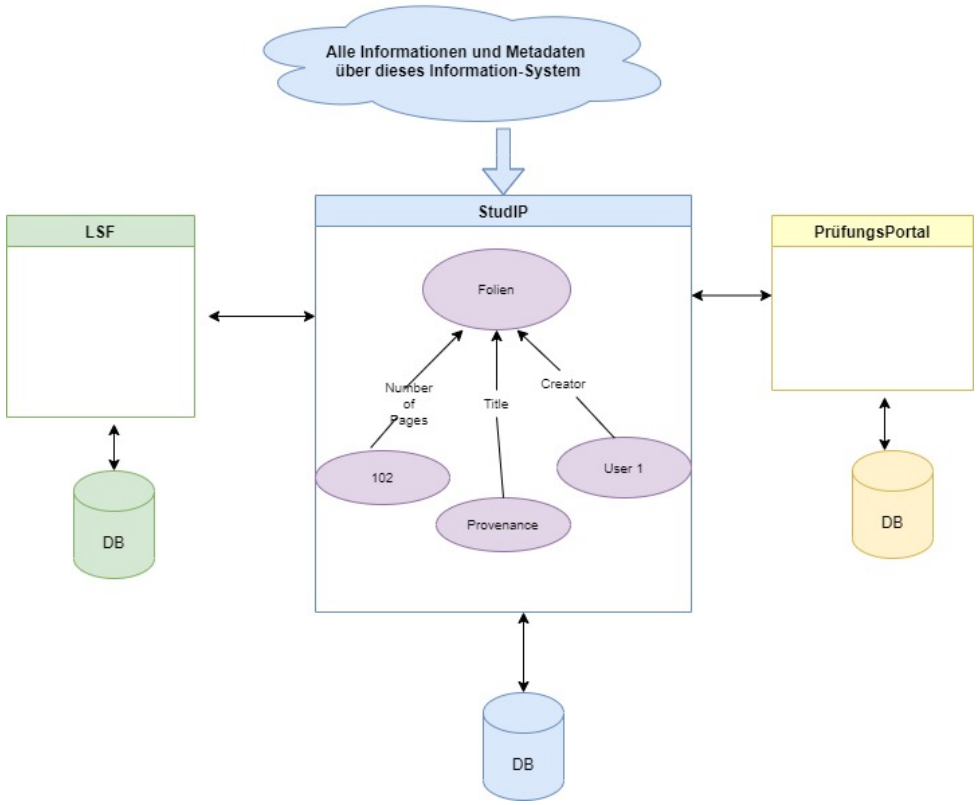

Abbildung 6.1: Information System Provenance im Studierendenbeispiel

7. Workflow Provenance

Die verschiedenen Aufgaben, die mit der Workflow-Provenance durchgeführt werden, decken drei verschiedene Dimensionen ab, die sich durch ihren Anwendungsbereich, ihre Granularität sowie die Art und Weise, wie die Provenance aufgezeichnet wird, unterscheiden. Der Anwendungsbereich bezieht sich auf die vier Bereiche Wissenschaft, Wirtschaft, Datenanalyse und allgemeine Programmierung. Provenance in Granularität bezieht sich auf zwei Bereiche: grobkörnige Provenance und feinkörnige Provenance. Die Form der Provenance konzentriert sich auf drei Bereiche: retrospektiv, prospektiv und evolutionär. In Abbildung 7.1 sind die drei genannten Dimensionen für die Workflow-Provenance zu sehen [HDB17].

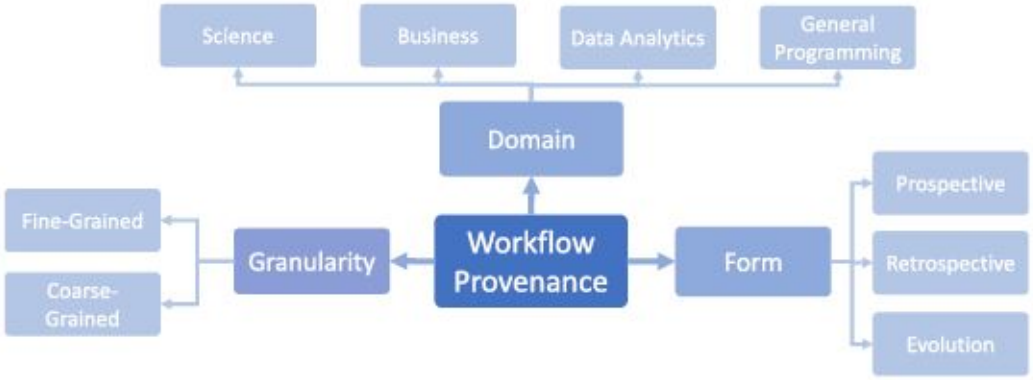

Abbildung 7.1: Drei verschiedene Dimensionen der Workflow-Provenance [HDB17]

Bevor auf die Besonderheiten der einzelnen Dimensionen eingegangen wird, können die allgemeinen Informationen über Input, Output und Prozesse (d. h. Workflows), die von der Workflow-Provenance abgedeckt werden, charakterisiert werden.

- ***Eingabe***: Die Provenance wird bei Workflow-Provenance-Lösungen in der Regel durch die Berücksichtigung der Eingabe erreicht. Die Eingabe ist wie folgt definiert: $I = (W, D, C)$. W stellt einen Arbeitsablauf dar, der ein gerichteter Graph ist, d. h. $W = (M, P)$, wobei M die Sammlung von Knoten ist, die die Workflow-Module darstellen, während die Kanten P die Abhängigkeiten zwischen den Modulen veranschaulichen. D bezieht sich auf Eingabedaten, seien es strukturierte relationale Daten, halb-strukturierte Daten oder unstrukturierte Daten. C ist ein Kontext, sodass der Workflow für die Verarbeitung der Eingabedaten wahrscheinlich ausführbare Parameter aus diesem Kontext verwendet wird.

- ***Ausgabe***: Der Output variiert je nach Anwendung der Provenance, der Input-Merkmale, der Art und der Granularität der Ausgabe von Provenance. Zum Beispiel könnte es sehr wohl feinkörnig sein, d. h. auf der Ebene der einzelnen Informationen, die von einem Arbeitsprozess aufbereitet werden, was der Ausgabe von Information-Provenance-Lösungen bestehender Ergebnisse entspricht. Im Gegensatz dazu könnte sich die Ausbeute auf Änderungen beschränken, die an der Definition des Arbeitsprozesses, an den Grenzen oder an anderen Informationen vorgenommen wurden.

- ***Workflow***: Graphenmodelle für Arbeitsabläufe, verschiedene und begrenzte Graphenmodelle sind enthalten. Beispielsweise können die Kanten P nach verschiedenen Bereichen definiert werden, sie können Daten- oder Kontrollabhängigkeiten darstellen. Graphen können auch auf nicht kreisförmige Graphen beschränkt werden. Im Folgenden wird ein allgemeines Graphenmodell betrachtet, damit keine Informationen über die Berechnungen eines Moduls $m \in M$ verfügbar sind. Daher benötigt m eine Eingabe I_m und einen Ausführungskontext C_m, um die Ausgabe O_m zu erzeugen. Das Ausführen eines Workflows führt zum Ausführen von Trace T. T ist ein gerichteter Graph, der nicht kreisförmig ist. Trace ist definiert als $T = (E, CD)$, sodass E sich auf die Ereignisse bezieht, die während des Workflows auftreten, d. h. die Aktivierung der Module im Workflow. Die zeitliche Abfolge der Aktivierung wird durch die CD im Workflow angezeigt [HDB17].

Im Folgenden werden Einzelheiten zu drei verschiedenen Dimensionen erörtert, die in Abbildung 7.1 und Tabelle 7.1 kurz dargestellt sind [HDB17].

Solution	Granularity		Form		
	Coarse	Fine	Prospective	Retrospective	Evolution
Science					
DFL [1]	✓	✓		✓	
Galaxy [84]	✓	✓		✓	
Kepler [6,32,126]	✓	✓	✓	✓	✓
Taverna [5,131,135]	✓	✓	✓	✓	
VisTrails [20,37,75]	✓			✓	✓
WebLab PROV [8]		✓		✓	
Business					
[59,125,171]	✓			✓	
[172]	✓			✓	
Data analytics					
Ariadne [82]	✓	✓		✓	
Differential DF [52]	✓	✓		✓	
HadoopProv [3]	✓	✓		✓	
Inspector Gadget [153]	✓	✓		✓	
Lipstick [9]	✓	✓		✓	
Newt [122]	✓	✓		✓	
RAMP [104]	✓	✓		✓	
Titian [106]	✓	✓		✓	
General programming					
Jif [144]	✓	✓	✓	✓	
LLVM/SPADE [79,175]	✓			✓	
noWorkflow [143,158]		✓		✓	
No+YesWorkflow [69,159]	✓	✓	✓	✓	
Pimentel et al. [160]	✓				✓
RDataTracker [120]	✓			✓	
Starflow [15]	✓		✓	✓	
YesWorkflow [127]	✓		✓		

Tabelle 7.1: Überblick über Workflow Provenance [HDB17]

7.1. Granularität

Die Ausgaben werden in verschiedenen Einzelheiten von Provenance-Lösungen bereitgestellt. Es wurden zwei verschiedene Arten von Granularität eingeführt, darunter grobkörnige Provenance und feinkörnige Provenance. Lösungen können aber auch eine Provenance zwischen den beiden erstellen. Informationen über die interne Funktionsweise von Workflow-Modulen werden in diesem Bereich als sehr wichtig angesehen. Wenn die interne Aktion der Workflow-Module nicht bekannt ist, werden diese Module als „Black Boxes“ bezeichnet, und wenn die internen Aktionsinformationen der Workflow-Module eindeutig bekannt sind, werden sie als „White Boxes“ bezeichnet [HDB17].

Grobkörnige Provenance Im Allgemeinen wird M als Black-Box-Modul eingeführt. Daher können Provenance-Lösungen zur Bereitstellung von Informationen über einzelne Daten, die von einem Workflow verarbeitet werden, nicht extrahiert werden. So werden die Ausgabedaten eines Moduls als abhängig von allen Eingaben eines Moduls und seinem Kontext betrachtet. Wenn die gesamte Ausgabe O_m eines Moduls $m \in M$ von der Eingabe I_m und dem vollständigen Ausführungskontext C_m abhängt, wird die Provenance im Allgemeinen als grobkörnig betrachtet und für alle Module gilt nach [HDB17]:

$$\forall m \in M : m \times I_m \times C_m \rightarrow O_m.$$

Feinkörnige Provenance Bei diesem Ansatz wird der Prozess der einzelnen Daten mithilfe von Provenance-Lösungen extrahiert. Diese Lösungen nutzen entweder das Merkmal des Prozesses oder die interne Funktion von White-Box-Modulen, um die genannten Daten zu extrahieren. Bei diesem Ansatz hängt die mit einem Workflow-Modul $m \in M$ verarbeitete Ausgabe $o \in O_m$ nicht von den gesamten Eingabedaten I_m und dem gesamten Ausführungskontext C_m ab, sondern von den Eingabeteilmengen $I'_m \subseteq I_m$ und Kontextteilmengen $C'_m \subseteq C_m$. Es gilt:

$$\forall m \in M,\ \forall o \in O_m : m \times I'_m \times C'_m \rightarrow O,$$

sodass I'_m und C'_m beide geringfügig sind [HDB17].

Beziehung zur Data Provenance Obwohl jede Daten mit feinkörniger Provenance zurückverfolgt werden kann, ist damit gegenwärtig nicht die Data Provenance gemeint. Wie in Abschnitt 3.1 dargelegt, bezieht sich die Data Provenance auf die spezifischste Art der Datenverarbeitung, sodass Anfragen auf der Grundlage von Operatoren mit expliziter Semantik bestimmt werden können. Nach der obigen Definition der feinkörnigen Workflow-Provenance wird nun deutlich, dass der Übergang zwischen Daten- und feinkörniger Workflow-Provenance fließend ist [HDB17].

7.2. Prospektive, retrospektive und evolutionäre Provenance

Die Workflow-Provenance deckt nicht nur verschiedene Arten von Provenance-Granularitäten ab, sondern wird auch in verschiedenen Formen definiert, wie z. B. prospektive Provenance, retrospektive Provenance und evolutionäre Provenance. Jede der drei Strukturen ist frei voneinander, sodass eine Herkunftsregelung nur eine, zwei oder alle drei unterstützen kann [HDB17].

Prospektive Provenance Die Struktur und der stationäre Kontext eines Arbeitsablaufs wird durch *prospektive Provenance* ermittelt. Dieser Ansatz ist nicht von der Ausführung des Workflows oder den Eingabedaten abhängig und wird von der Workflow-Bestimmung W und dem Teil der vorbestimmten Parameter von C abgeleitet. Dieser Ansatz wird verwendet, um eine abstrakte Zusammenfassung eines Arbeitsablaufs in verschiedenen Bereichen zu erstellen [HDB17].

Retrospektive Provenance Die Informationen über die Ausführung eines Workflows beziehen sich auf die *retrospektive Provenance*, d. h. Informationen, die während der Ausführung des Work-

flows gewonnen werden. Retrospektive Provenance wird durch Provenance-Lösungen erhalten, die Zugriff auf alle Eingabeparameter $I = (W, D, C)$ haben. Die retrospektive Provenance wird durch Werkzeuge des Arbeitsablaufs oder seiner Ausführungsumgebung erfasst. Die Erfassung einer Ausführungsspur T während der Workflow-Ausführung wird ermöglicht. Die Spur T wird regelmäßig zu einem wesentlichen Bestandteil des Ergebnisses der Provenance-Lösung [HDB17].

Evolutionäre Provenance Die Variationen, die zwischen zwei Verschreibungen der Eingabe I und I' entstehen, werden durch die *evolutionäre Provenance* dargestellt. Diese Änderungen können auf W, D oder C angewandt werden. Immer wenn mindestens eine dieser Änderungen vorgenommen wird, können diese Änderungen durch die Provenance-Lösung überwacht werden. Schnelle Duplikate bei verschiedenen Daten, Parametern und Workflow-Manipulationen werden durch Evolution Provenance reduziert. „Diese Form der Provenance wird im Bereich der wissenschaftlichen Arbeitsabläufe auch als Provenance des Prozesses bezeichnet“ [HDB17].

7.3. Anwendungsbereiche

Die Workflow-Provenance wurde in verschiedenen Bereichen wie der experimentellen Wissenschaft, der Wirtschaft, der Datenanalyse und der allgemeinen Programmierung eingesetzt [HDB17]. Im Folgenden wird auf die genannten Anwendungsbereiche eingegangen.

7.3.1. Wissenschaftliche Arbeitsabläufe

Wissenschaftliche Arbeitsabläufe werden in der Regel in wissenschaftlichen Workflow-Management-Systemen (SWfMS) erstellt, verwaltet und durchgeführt. Wissenschaftliche Workflows sind für zahlreiche wissenschaftliche Anwendungen erschienen. Galaxy, Taverna und Kepler werden zum Beispiel bei Untersuchungen zu biomedizinischen Themen wie Genomforschung oder Biowissenschaften eingesetzt (wie in Tabelle 7.1 dargestellt) [HDB17].

Merkmale des Arbeitsablaufs Wissenschaftliche Arbeitsabläufe wurden zunächst nach einer datengesteuerten Methode erstellt. Die Datenabhängigkeiten werden durch die Kanten P des Workflow-Graphen definiert, was zu nichtzyklischen Graphen führt, andererseits erlauben einige SWfMSs ebenfalls, Kontrollströme in gewisser Weise darzustellen (z. B. Kontrollschleifen), sodass die entstehenden Arbeitsabläufe Zyklen enthalten können [HDB17].

Granularität Die Granularität im Bereich der wissenschaftlichen Arbeitsabläufe wird sich auf die Erfassung der grobkörnigen Provenance konzentrieren [HDB17].

Form Vielfältige Provenance-Lösungen im wissenschaftlichen Bereich erlauben es, retrospektive Provenance auf jeder Granularität zu erfassen. Ihre Ausgabe ist in der Regel ein Provenance-Graph, der von der Ausführungsspur T abhängt. Die Erfassung der prospektiven Provenance wird auch durch andere Lösungen unterstützt. Zusätzlich können Provenance-Lösungen nicht nur auf statische Workflow-Untersuchungen, sondern auch auf Annotationen angewiesen sein. Zum Beispiel wird Evolution Provenance von Kepler unterstützt [HDB17]. Dieses Tool wird genauer in Kapitel 9.4 vorgestellt.

7.3.2. Geschäftliche Arbeitsabläufe

Prozesse zwischen und innerhalb von Unternehmen werden durch geschäftliche Arbeitsabläufe unterstützt.

Merkmale des Arbeitsablaufs Mehrere Aktivitäten, Aufgaben oder Ereignisse sind in Geschäftsabläufe eingebunden, die möglicherweise menschliche Interaktion erfordern. Sie werden durch den Kontrollfluss gesteuert und unterscheiden sich daher von wissenschaftlichen Arbeitsabläufen. Daher wird der Kontrollfluss durch die Kanten P eines Geschäftsablaufs W repräsentiert. In dieser Anwendung wird der Datenfluss in der Regel nicht explizit modelliert [HDB17].

Granularität Die Sammelstellen für die Provenance und die Erstellung eines gemeinsamen Modells von Provenance wird als eine Herausforderung bei der Erfassung der Provenance in diesem Anwendungsbereich erkannt. Daher kann es erforderlich sein, dass sich aus der Ausführung von Workflow-Modulen Informationsabhängigkeiten ergeben und wird es nicht möglich sein, den Ursprung und den Ableitungspfad jedes Datenelements in jeder Datei zu verfolgen. Die meisten Geschäftsabläufe werden aus Webdiensten aufgebaut, sodass die Interna nicht bekannt sind und eine feinkörnige Herkunftsverfolgung nicht möglich ist [HDB17].

Form Weil das Ziel der Provenance bei diesem Ansatz die Einhaltung von Vorschriften, die Rechenschaftspflicht oder die Vertrauenswürdigkeit ist, liegt der Schwerpunkt auf der retrospektiven Provenance [HDB17].

7.3.3. Daten-Analytik

Datenanalyse-Motoren sind der nächster Anwendungsbereich. Sie werden sich von wissenschaftlichen Workflow-Systemen und Tools zur Verwaltung von Geschäftsprozessen unterscheiden. Durch diesen Ansatz werden strukturierte oder unstrukturierte Daten aus heterogenen Quellen verarbeitet [HDB17].

Merkmale des Arbeitsablaufs Die Arbeitsabläufe der Informationsanalyse sind datengesteuert. Die Datenabhängigkeiten zwischen den Datenverarbeitungs-Modulen M werden durch die Kanten P verbunden und der Workflow-Graph W ist nicht zyklisch [HDB17].

Granularität Für Datenanalysesysteme wird die feinkörnige Provenance durch Provenance-Lösungen bereitgestellt, die für diese Systeme auch die grobkörnige Provenance umfassen. Der Ableitungspfad der einzelnen Datenelemente wird nach diesem Ansatz berechnet [HDB17].

Form Bei der Ausführung von Arbeitsabläufen in Datenanalysesystemen wird die retrospektive Provenance verwendet [HDB17].

7.3.4. Allgemeine Programmierung

Provenance-Lösungen für universell einsetzbare Programmiersprachen wie C++, Java oder Python sind in den letzten Jahren zum Gegenstand der Forschung geworden. Anders als z. B.

SWfMS verfügen diese Sprachen nicht über eine intrinsische Provenance-Unterstützung. Infolgedessen werden zahlreiche verschiedene Möglichkeiten zum Umgang mit der Provenance erforscht [HDB17].

Merkmale des Arbeitsablaufs Aus Programmen, die in allgemeinen Programmiersprachen geschrieben sind, werden Workflow-Graphen W genommen. Die Kanten P zeigen Datenabhängigkeiten und Kontrollabhängigkeiten und diese Art von Workflow ist ein Zyklus [HDB17].

Granularität Für Programmiersprachen können sowohl feinkörnige als auch grobkörnige Provenance-Lösungen verwendet werden [HDB17].

Form In Bezug auf Programmiersprachen werden alle Arten von Provenance untersucht [HDB17].

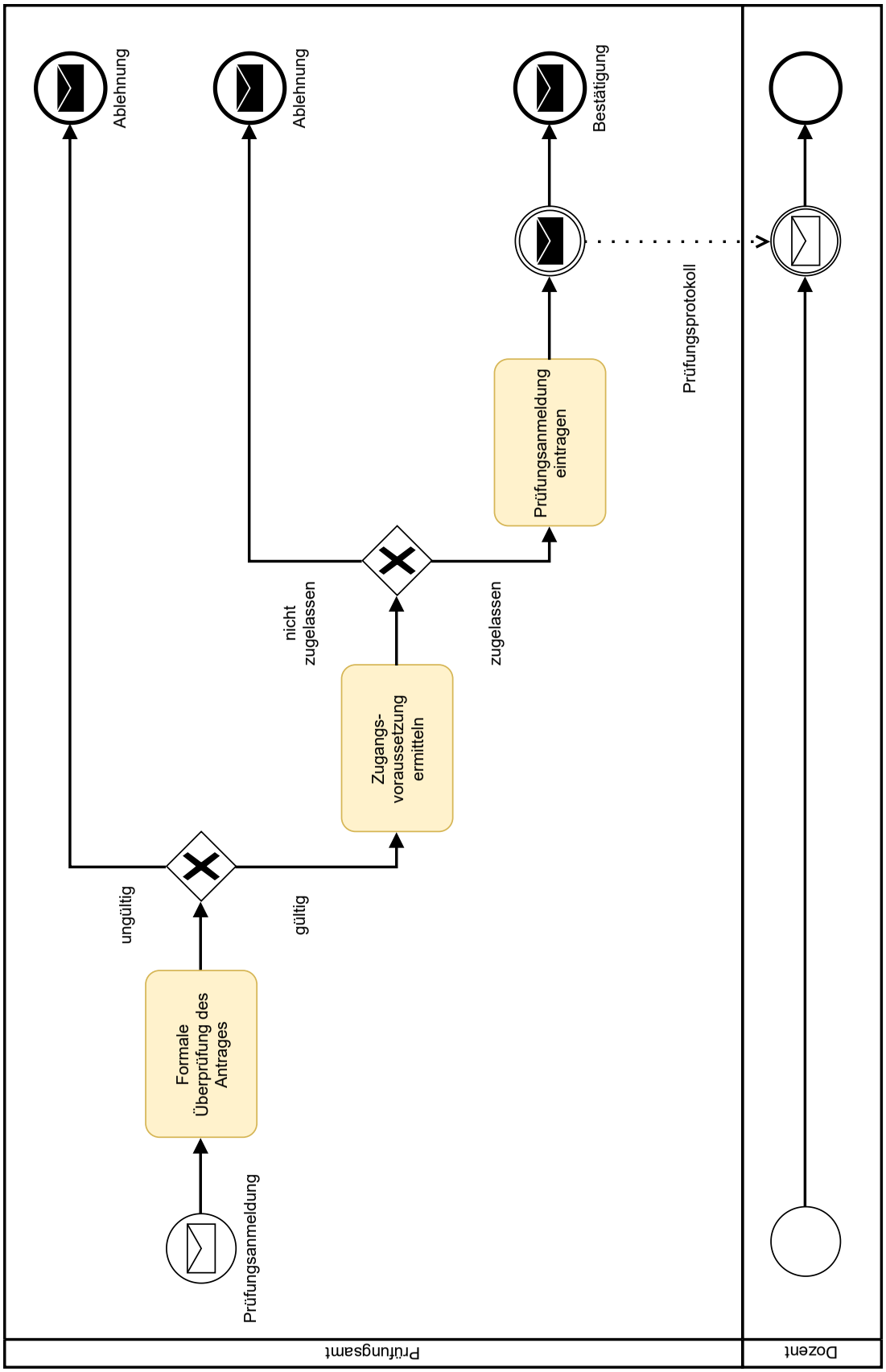

Abbildung 7.2: Workflow einer Prüfungsanmeldung in BPMN

7.4. Beispiel in BPMN

Was ist BPMN? Business Process Model and Notation (BPMN) [Obj14; Obj13] ist eine Beschreibungssprache zur Beschreibung von Workflows. Diese wird von der Object Management Group standardisiert. Dokumente sind grafische Repräsentationen von Arbeitsabläufen, welche durch Events unterschiedlichster Arten ausgelöst werden. Es lassen sich Sequenzen von Aufgaben, Bedingungen und Schleifen darstellen. Zu den weiteren Modellierungsmöglichkeiten gehören Organisationen, der Austausch von Nachrichten sowie Datenspeicher. Die aktuellste Version 2.0.2 wurde im Jahr 2014 veröffentlicht.

Studierendenbeispiel in BPMN Wir wollen uns überlegen, wie eine Prüfungsanmeldung an unserer ausgedachten Universität ablaufen könnte. Hierzu zeigt die Abbildung 7.2 den Ablauf in BPMN. Zuerst geht eine Prüfungsanmeldung als Nachricht bei dem Prüfungsamt ein. Dadurch wird der Workflow aktiv und das Prüfungsamt beginnt mit der ersten Aufgabe: der formalen Überprüfung des Antrages. Zwei Bedingungen müssen erfüllt sein, damit der Antrag gültig ist. Erstens muss der Antrag ordnungsgemäß ausgefüllt, also Felder wie Name oder Studiengang eingetragen sein. Zweitens muss die Prüfung, welche der Studierende anmelden möchte, auch in seiner Studienordnung vorgeschrieben sein. Ein Studierender des Faches „Informatik“ darf sich beispielsweise in „Datenbanken“ prüfen lassen, aber nicht in „Epochen der Alten Geschichte“. Die Formalitäten sind erfüllt. Als Nächstes wird die Zugangsvoraussetzung ermittelt. Es ist für das Modul vorgeschrieben, dass mindestens 75 % der Punkte in den Hausaufgaben erreicht werden müssen, um zur Prüfung zugelassen zu werden. Auch hier ist alles in Ordnung. Die verantwortliche Person im Prüfungsamt legt den Antrag zu den Unterlagen und trägt die angemeldete Prüfung im System ein. Ein Prüfungsprotokoll für das Modul wird erstellt und an den Dozenten geschickt. Modelliert wird dieser Vorgang als Zwischenereignis. Der gestrichene Pfeil zeigt an, dass es sich hierbei nicht um eine Sequenz von Aktivitäten handelt, sondern um einen Datenfluss. Nun wird der Dozent aktiv und nimmt das Prüfungsprotokoll entgegen. Seine Aufgabe bei der Prüfungsanmeldung besteht ausschließlich darin, das Protokoll entgegenzunehmen. Zuletzt schickt das Prüfungsamt eine Nachricht über die erfolgreiche Registrierung zu dem Studierenden und der Workflow endet. Es gibt noch die Fälle, dass der Antrag nicht formgemäß ausgefüllt ist oder der Studierende nicht zugelassen ist. In der Abbildung sind zwei exklusive Gateways eingezeichnet, erkennbar an dem großen „X“. Ist der Antrag nicht ordnungsgemäß ausgefüllt, wird er abgelehnt und der Studierende muss einen neuen Antrag stellen. Wenn die Zugangsvoraussetzung nicht erfüllt ist, wird der Antrag ebenfalls abgelehnt und eine Registrierung ist erst wieder im nächsten Semester möglich. Beide Gateways münden jeweils in eine Nachricht an den Antragsteller, dass die Anmeldung nicht erfolgt ist. Damit endet der Workflow.

Später wird dieses Beispiel in PROV nachmodelliert, Details finden sich im Kapitel 10.2.

8. Data Provenance

Bei der Data Provenance handelt es sich, wie bereits in dem Abschnitt 3.1 vorgestellt, um die spezifischste Provenanceart. Die Data Provenance beantwortet die Frage, welche Daten für die Beantwortung benötigt wurden. Im Allgemeinen gibt es vier Arten von Fragen in der Data Provenance: ***how***, ***why***, ***why-not***, ***where***. Die Antwort auf eine Frage der Data Provenance lässt sich in eine von vier Kategorien einteilen. Die Arten der Fragen und Antworten in der Data Provenance werden im Abschnitt 8.1 vorgestellt. Danach wird im Abschnitt 8.2 die Provenance-Halbringe dargestellt. Diese lassen sich nutzen, um die Data Provenance zu berechnen.

8.1. Anfragen und Antworten

Die vier Anfragen der Data Provenance sind:

- ***where***: Welche Tabellen oder Attribute wurden für die Anfrage verwendet?
- ***why***: Welche Tupel sind relevant für das Ergebnis gewesen?
- ***why-not***: Warum fehlt ein bestimmtes Element im Ergebnis?
- ***how***: Wie ist das Ergebnis zustande gekommen?

In [Heu16] werden vier Provenance-Antworten dargestellt. In dem Artikel [HDB17] findet sich für die ***why-not*** außerdem noch eine instanzbasierte Antwort. Die Antworten lauten:

- **Extensionale Antwort**: Es werden die Ausgangsdaten ausgegeben, z. B. durch eine Liste an Tupeln.
- **Intensionale Antwort**: Die Antwort beschreibt, z. B. in textueller Form, die Daten.
- **Anfragebasierte Antwort**: Es werden die Selektionsprädikate als Antwort ausgegeben.
- **Modifikationsbasierte Antwort**: Eine minimale Änderung der Anfrage auf die Datenbank wird vorgeschlagen, damit ein gewünschtes Ergebnis enthalten ist.
- **Instanzbasierte Antwort**: Bei einer ***why-not***-Anfrage wird eine Änderung oder ein möglicher neuer Eintrag für die Datenbank vorgeschlagen.

Nicht jede Provenanceanfrage kann mit allen Antwortarten beantwortet werden. So ist eine modifikationsbasierte Antwort nur für eine ***why-not***-Anfrage sinnvoll. ***Where***, ***why*** und ***how*** werden meist durch eine extensionale Antwort beantworten. Diese drei Anfragen lassen sich folgend nach ihrem Informationsgehalt ordnen: ***where*** $\preceq$ ***why*** $\preceq$ ***how***. Daraus folgt, dass sich aus einer Antwort auf ***how*** auch eine Antwort auf ***why*** und ***where*** erstellen lässt.

In den nächsten vier Abschnitten des Kapitels werden die Anfragen genauer vorgestellt und im Abschnitt 8.2 werden die Provenance-Halbringe vorgestellt, welche sich zur Berechnung der Data Provenance eignen. Im Abschnitt 8.2.3 wird die Provenance für einfache Aggregationen dargestellt.

8.1.1. *How*-Provenance

Die ***how***-Provenance beantwortet Fragen wie „Wie ist das Ergebnistupel entstanden?“ und „Warum gibt es diese Ergebnistupel?“ Als Antwort werden Provenance-Polynome verwendet, welche angeben, wie einzelne Tupel in das Ergebnistupel eingeflossen sind. Die Provenance-Polynome basieren auf den in dem Abschnitt 8.2 vorgestellten Provenance-Halbringen. In Abschnitt 8.2.2 werden die Polynome noch einmal genau vorgestellt.

Beispiel In der zweiten Anfrage des Studierendenbeispiels (Anfrage 2.2) wird die Anfrage gestellt, welche Studierenden am Kurs mit der Nummer 005 teilnehmen. In der Tabelle 8.1 ist das Ergebnis dieser Anfrage mit der ***how***-Provenance zu sehen. Als Provenance werden die Polynome $S_4 \cdot P_{16}$ und $S_7 \cdot P_{17}$ angegeben. Diese bedeuten, dass die Tupel jeweils einmal in das Ergebnis eingehen und durch einen Verbund kombiniert wurden.

s_id	lastname	firstname	p_id	course_nr	***how***
S_4	Harrison	Elisabeth	P_{16}	005	$S_4 \cdot P_{16}$
S_7	Smith	Jack	P_{17}	005	$S_7 \cdot P_{17}$

Tabelle 8.1: Ergebnis der Anfrage 2.2 mit ***how***-Provenance

8.1.2. *Why*-Proveanance

Die ***why***-Provenance gibt an, warum ein Ergebnis entstanden ist. Hierzu werden in [CWW00] sogenannte Zeugen eingeführt. Ein Zeuge ist eine Menge von Tupeln der Ursprungsrelation, welche ausreichen, damit die Anfrage das Ergebnis erzeugt. Wenn mehrere Zeugen für ein Ergebnis vorhanden sind, werden diese in einer Zeugenmenge zusammengefasst. Da die Zeugen auch nicht relevante Tupel enthalten können, kann eine Zeugenmenge schnell wachsen. Dieses Problem wird in [BKT01] gelöst. In der Arbeit [Aug17] wird ein Überblick über die Konzepte gegeben.

Definition 8.1 (Zeugenmenge)**.**
Sei d eine Datenbank, Q eine Anfrage, so ist der Zeuge eines Ergebnistupels $t \in Q(d)$ eine Teilrelation $w \subseteq d$ mit $t \in Q(w)$. Wenn ein Tupel $k \in \mathbb{N}$ Zeugen hat, so ist die Zeugenmenge W gegeben durch $W = \{w_1, \ldots, w_k\}$.

Eine Zeugenbasis fast alle Zeugenmengen einer Ergebnisrelation zusammen.

Definition 8.2 (Zeugenbasis)**.**
Eine Zeugenbasis $W_{Q,d(t)}$ ist die Menge aller Zeugenmengen, das bedeuten, dass $W_{Q,d(t)} = \{W_1, \ldots, W_n\}$ mit $W_i = \{w_{i1}, \ldots, w_{ik}\}$ und $1 \leq i \leq n, k \in \mathbb{N}$.

Aufbauend auf der Zeugenbasis kann eine minimale Zeugenbasis definiert werden. Diese ist bei zwei äquivalenten Anfragen gleich.

Definition 8.3 (Minimale Zeugenbasis)**.**
Eine Zeugenbasis $W_{Q,d(t)}$ für Q ist minimal genau dann, wenn keine echte Teilmenge von $W_{Q,d(t)}$ selber eine Zeugenbasis für Q ist.

Beispiel In der Beispielanfrage 2.2 wird nach den Besuchern der Veranstaltung mit der Kursnummer 005 gesucht. Die Zeugenbasis der Ergebnisrelation (Tabelle 2.2) der Anfrage ist:

$$W = \{\{S_4, P_{16}\}, \{S_7, P_{17}\}\}$$

Aus der ***how***-Provenance der einzelnen Ergebnistupel lässt sich die ***why***-Provenance erkennen. Die Annotationen der einzelnen Provenancepolynome tauchen bei ***why*** jeweils zusammen in einer Menge auf.

8.1.3. *Where*-Provenance

Die ***where***-Provenance gibt an, woher die Daten für ein Ergebnis kommen. Sie beschreibt den Zusammenhang von Ursprungsort und Ergebnis. Im Gegensatz zur ***how***- und ***why***-Provenance, welche mit Tupelidentifikatoren arbeiten, kann bei der ***where***-Provenance auch mit Relationsnamen gearbeitet werden. Für eine Anfrage Q über der Datenbank $d = \{r_1, \ldots, r_n\}$ ist die ***where***-Provenance für $t \in Q(d)$ durch die Menge der Ursprungsorte $r_t = \{r_i \mid r_i \text{ ist Quellrelation von } t \text{ und } i \in \{1, \ldots, n\}\}$ gegeben. Wenn die ***where***-Provenance auf Relationsebene betrachtet wird, werden nur die Relationen ausgegeben, aus welchen Attribute ins Ergebnis übernommen wurden. Relationen, welche für die Anfrage benötigt werden, verschwinden, wenn deren Attribute wegprojiziert werden.

Beispiel Die ***where***-Provenance auf Relationsebene der Beispielanfrage 2.2 ist für die beiden Ergebnistupel die Menge {STUDENTS, PARTICIPANTS}, da Attribute aus diesen beiden Tabelle im Ergebnis erscheinen.

8.1.4. *Why-not*-Provenance

Die ***why-not***-Provenance stellt die Frage, warum ein Tupel nicht im Ergebnis ist. Dies zu fragen kann sinnvoll sein, wenn ein bestimmtes Ergebnis erwartet wird. Die Antwort einer ***why-not***-Frage kann helfen eine Anfrage besser zu formulieren. Die für die ***how***-, ***why***- und ***where***-Provenance genutzte extensionale Antwort eigenet sich nicht für die ***why-not***-Provenance. Stattdessen können anfragebasierte, modifikationsbasierte oder instanzbasierte Antworten verwendet werden. Doch nicht für jede ***why-not***-Anfrage kann automatisch eine Antwort generiert werden [Her15].

Beispiel Der Student DONALD MOORE bewirbt sich bei einem Lehrstuhl als Korrektor der Hausaufgaben für den Kurs mit der Nummer 001. Der Lehrstuhl möchte, dass ein Korrektor das Fach mit einer guten Note abgeschlossen hat und stellt mit der Anfrage 8.1 eine Liste mit infrage kommenden Studierenden auf.

```
SELECT student_id, grade
FROM grades
WHERE grade < 2
AND course_nr = '001';
```

Anfrage 8.1: Beispielanfrage für die ***why-not***-Provenance

Der Student fragt sich nun, warum er, beziehungsweise seine STUDENT_ID (001) nicht im Ergebnis vorkommt. Als Antwort sind drei Arten möglich:

- Die **anfragebasierte Antwort** gibt an, durch welches Selektionsprädikat das gewünschte Tupel aus dem Ergebnis entfernt wurde. In diesem Beispiel ist dies die Bedingung GRADE < 2, weil der Student in diesem Modul eine 2,0 als Note hat.
- Die **modifikationsbasierte Antwort** baut auf der anfragebasierten Antwort auf. Für das gefundene Selectionsprädikat wird eine Veränderung vorgeschlagen. Hier z. B. die Ersetzung von GRADE < 2 durch GRADE $=< 2$, damit der Student mit der Note 2.0 noch im Ergebnis enthalten ist.
- Die **instanzbasierte Antwort** gibt aus, welches Tupel in dem Datensatz fehlt, damit das erwartete Tupel im Ergebnis enthalten ist. Hier wäre z. B. möglich, dass ein Prüfungseintrag mit einer Note kleiner als zwei vorgeschlagen wird. Für dieses Beispiel ist diese Antwort nur bedingt hilfreich.

8.2. Provenance-Halbringe

Die Provenance-Halbringe bilden die theoretischen Grundlagen für die Provenance-Polynome. Welche für die ***how***-Provenance verwendet werden. In dem Abschnit 8.2.2 werden die Provenance-Polynome mit einem Beispiel dargestellt. Sie wurden in dem Artikel [GKT07] vorgestellt. Der neueren Artikel [GT17] gibt mehr einen Überblick und eignet sich hierdurch als guter Anfang für weitere Literaturarbeit zum Thema.

Definition 8.4 (Kommutativer Halbring [GT17])**.**
Eine Struktur $(K,\ +_K,\ \cdot_K,\ 0_K,\ 1_K)$ ist ein kommutativer Halbring, wenn $(K, +_K, 0_K)$ und $(K, \cdot_K, 1_K)$ kommutative Monoide sind, $\cdot_K$ distributiv über $+_K$ und $0_K \cdot a = a \cdot 0_K = 0_K$ gilt.

Für die Provenance sind unter anderem der Halbring, der natürlichen Zahlen $(\mathbb{N}, +, \cdot, 0, 1)$ und der boolesche Halbring $(\mathbb{B}, \vee, \wedge, \bot, \top)$. Für die Provenancepolynome wird später der Halbring $(\mathbb{N}[X], +, \cdot, 0, 1)$ benötigt.

Definition 8.5 (Träger einer Funktion [GT17])**.**
Sei D eine Menge und A eine Funktion $A : D \to K$ so wird der Träger von A definiert als:

$$supp(A) = \{x \in D \mid A(x) \neq 0\}$$

In dem Artikel [GKT07] werden für die Definition von K-Relationen Tupel als Funktion $t : U \to D$ verstanden, also als eine Abbildung vom Universum auf eine Domäne von Werten. Die Domäne D wird für die Definition unveränderbar gemacht und die Menge aller solcher Tupel wird als **U**-Tup bezeichnet. Relationen über U sind dann Teilmengen von **U**-Tup.

Definition 8.6 (K-Relation [GKT07])**.**
Sei K ein kommutativer Halbring mit einem ausgezeichneten Element 0 und U eine endliche Menge von Attributen, so ist eine K-Relation definiert als Funktion ρ : **U**-Tup $\to K$, sodass $supp(\rho) = \{t \mid \rho(t) \neq 0\}$.

In einer K-Relation wird also **U**-Tup durch die Funktion ρ auf einen kommutativen Halbring K abgebildet. Der Träger von ρ beinhaltet alle in einer Relation vorhandenen Elemente. Elemente aus **U**-Tup, welche zwar möglich sind, aber nicht in einer Relation sind, werden auf das Nullelement von K abgebildet.
In einer K-Relation werden also die Einträge einer Relation auf Werte des Halbringes abgebildet. Die Werte des Halbringes werden als Annotation der Tupel aufgefasst. In der Tabelle STUDENTS des Studierendenbeispiels sind die Annotationen in der Spalte S_ID dargestellt.

8.2.1. K-relationale Algebra

Die K-relationale Algebra beschreibt, wie mit den Annotationen umgegangen wird, wenn Operationen über K-Relationen ausgeführt werden. Der +-Operator wird für die Vereinigung und Projektion verwendet, wenn zwei annotierte Tupel zusammengefasst werden. Der ·-Operator wird für den natürlichen Verbund verwendet, um Annotationen von Tupeln zu kombinieren. Die beiden Werte 1 und 0 werden als Annotation für „in“ beziehungsweise als „außerhalb“ einer Relation interpretiert.

Definition 8.7 (Positive K-relationalen Algebra [GKT07])**.**
Sei $(K, +_K, \cdot_K, 0, 1)$ eine algebraische Struktur mit zwei binären Operationen $+_K$ und $\cdot_K$, sowie zwei ausgezeichneten Elemente 0 und 1 so wird die positive K-relationale Algebra folgend definiert:

- **Leere Relation**: Für jede Attributmenge U existiert $\emptyset$: **U**-Tup $\to K$, sodass $\emptyset(t) = 0$
- **Vereinigung**: Sei R_i : **U**-Tup $\to K$ mit $i = 1, 2$ gegeben, so wird $R_1 \cup R_2$: **U**-Tup $\to K$ definiert durch:

$$(R_1 \cup R_2)(t) := R_1(t) +_K R_2(t)$$

- **Projektion**: Sei $R : \mathbf{U}\text{-Tup} \to K$ und $V \subseteq U$, dann ist $\pi_V R : \mathbf{V}\text{-Tup} \to K$ definiert durch:

$$(\pi_V R)(t) := \sum_{t=t' \text{ über } V \text{ und } R(t')\neq 0} R(t')$$

  mit $+_K$ für die Summierung über $t' \in supp(R)$.

- **Selektion**: Sei $R : \mathbf{U}\text{-Tup} \to K$ und ein Selektionsprädikat $\mathbf{P}$, welches jedem $\mathbf{U}$-Tup 0 oder 1 zuordnet, dann ist $\sigma_{\mathbf{P}} R : \mathbf{U}\text{-Tup} \to K$ gegeben durch:

$$(\sigma_{\mathbf{P}} R)(t) := R(t) \cdot_K \mathbf{P}(t)$$

- **Natürlicher Verbund**: Sei $R_i : \mathbf{U}_i\text{-Tup} \to K$ mit $i = 1, 2$, dann ist $R_1 \bowtie R_2$ eine K-Relation über $U_1 \cup U_2$ definiert durch:

$$(R_1 \bowtie R_2)(t) := R_1(t_1) \cdot_K R_2(t_2)$$

  mit $t_1 = t$ auf U_1 und $t_2 = t$ auf U_2.

- **Umbenennung**: Die Umbenennung $\rho_\beta R$ mit $R : \mathbf{U}\text{-Tup} \to K$ und $\beta : U \to U'$ ist eine K-Relation über U' und gegeben durch:

$$(\rho_\beta R)(t) := R(t \circ \beta)$$

8.2.2. Provenance-Polynome

Die Basis für Provenance-Polynome ist der Halbring $(\mathbb{N}[X], +, \cdot, 0, 1)$. Dieser besteht aus Polynomen mit Variablen aus X und Koeffizienten aus $\mathbb{N}$. Die Menge X ist die Menge aller Tupelidentifikatoren einer Datenbankinstanz. Im Folgendem soll an einem kleinen Beispiel die Berechnung der Provenance für eine Anfrage gezeigt werden.

Als Beispielanfrage wird die Fragestellung „Für welche Kurse wurden im SS 16 Prüfungen durchgeführt?“ betrachtet. Für die Beantwortung ist es notwendig, die Tabellen COURSES und GRADES zu verbinden, nach dem Semester zu selektieren und auf den Veranstaltungsnamen zu projizieren. In Anfrage 8.2 ist die Anfrage in SQL dargestellt.

```
SELECT c.title, g.course_nr
FROM courses c, grades g
WHERE c.course_nr = g.course_nr
AND semester = 'SS 16';
```

Anfrage 8.2: Beispielanfrage für Provenance-Polynome

Da für die Berechnung die beiden Tabellen COURSES und GRADES benötigt werden, besteht die Menge der Tupelidentifikatoren X aus $C_1, \ldots, C_9$ und $G_1 \ldots, G_{23}$. Die Tabelle GRADES

course_nr	$\mathbb{N}[X]$
001	$G_1 + G_2 + G_3 + G_4$
007	$G_{19} + G_{20}$
009	$G_{21} + G_{23}$

Tabelle 8.2: Ergebnis der Teilanfrage $\pi_{course\_nr}(\sigma_{semester=\text{SS }16}(Grades))$ mit Annotation

nach der Selektion und Projektion ist in der Tabelle 8.2 dargestellt. In der Spalte $\mathbb{N}[X]$ sind die Provenance-Polynome der Tupel aus dem Zwischenergebnis zu sehen. Da mehrere Tupel zu dem gleichen Ergebnis führen, werden diese zusammengefasst und die Annotation durch die Addition verbunden.

title	*course_nr*	$\mathbb{N}[X]$
Database Systems and Data Science	001	$(G_1 + G_2 + G_3 + G_4) \cdot C_1$
Law and Computer Science	007	$(G_{19} + G_{20}) \cdot C_7$
Networks ans Cybersecurity	009	$(G_{21} + G_{23}) \cdot C_9$

Tabelle 8.3: Ergebnis der Beispielanfrage mit den Provenance-Polynomen

Das Ergebnis der Anfrage ist in der Tabelle 8.3 dargestellt. In der Spalte mit den Provenance-Polynomen ist zu erkennen, dass die Polynome der verbundenen Tupel durch die Multiplikation verbunden wurden. Die Darstellung der Polynome beinhaltet hier eine Klammerung. Durch Ausmultiplizieren kann diese aufgelöst werden. Der Ausdruck $(G_{21} + G_{23}) \cdot C_9$ kann zum Beispiel auch als $G_{21} \cdot C_9 + G_{23} \cdot C_9$ geschrieben werden.

8.2.3. Aggregation

In dem Artikel [ADT11] wird Provenance für Aggregation- und Gruppierungs-Operationen vorgestellt. Im Folgenden wird die Provenance für einfache Aggregationen, also solche, die in der Anfrage als letztes durchgeführt werden, dargestellt. Die Provenance wird definiert für Aggregationen, welche durch einen kommutativen Monoide definiert sind. Beispiele hierfür sind:

- SUM $= (\mathbb{R}, +, 0)$
- MIN $= (\mathbb{R}^{+/-\infty}, min, +\infty)$
- MAX $= (\mathbb{R}^{+/-\infty}, max, -\infty)$
- PROD $= (\mathbb{R}, \times, 1)$.

Die Provenance für Aggregationen arbeitet mit der Bag-Semantik (auf deutsch Multimengen-Semantik). Das heißt, dass keine Duplikateliminierung stattfindet. Die Häufigkeit der Tupel kann durch eine Annotation vermerkt werden. Ein Beispiel für die Bag-Semantik ist in der Tabelle 8.4 gegeben. Die, in der Provenance für Aggregationen vorkommenden, Annotationen aus $\mathbb{N}[X]$ werden als Platzhalter für die Häufigkeit der einzelnen Tupel verstanden. In den meisten Fällen ist die Häufigkeit 1, wie z. B. in diesem Studierendenbeispiel. In dem Beispiel 3.4 des Artikels [ADT11] kommen aber auch Tupel mit anderen Häufigkeiten vor.

fullname
Professor A
Professor B
Professor C
Professor D
Professor E
Lecturer A
Lecturer B

fullname	$\mathbb{N}$
Professor A	3
Professor B	2
Professor C	1
Professor D	2
Professor E	1
Lecturer A	1
Lecturer B	1

Tabelle 8.4: Ergebnis der Anfrage $\pi_{fullname}(Lectures)$. Auf der linken Seite ist das Ergebnis in der Mengen-Semantik und auf der rechten in der Bag-Semantik, mit der Häufigkeit in der Spalte $\mathbb{N}$, zu sehen.

Bei der Provenance für Aggregationen werden nicht nur die verwendeten Tupel angegeben, stattdessen wird die Berechnung des Werts beschrieben. Für die Umsetzung wird das K-Halbmodul eingeführt, welches den Monoiden der Aggregation um eine binäre Operation erweitert. Diese „verbindet“ die Aggregation mit der Provenance.

Definition 8.8 (K-Halbmodul [ADT11])**.**
Sei K ein kommutativer Semiring und $(W, +_W, 0_W)$ ein kommutativer Monoid, so ist die Struktur $(W, +_W, 0_W, *_W)$ mit $*_w : K \times W \to W$ ein K-Halbmodul, sodass für alle $k, k_1, k_2 \in K$ und w, w_1, w_2 gilt:

- $k \star_W (w_1 +_W w_2) = k \star_W w_1 +_W k \star_W w_2$
- $k \star_W 0_W = 0_W$
- $(k_1 +_K k_2) \star_W w = k_1 \star_W w +_W k_2 \star_W w$
- $0_K \star_W w = 0_W$
- $(k_1 \cdot_K k_2) \star_W w = k_1 \star_W (k_2 \star_W w)$
- $1_K \star_W w = w$

Außerdem kann für jeden kommutativen Monoid $W, +_w, 0_w$ auch $nx = x +_w \cdots +_w x$ (n mal) mit $n \in \mathbb{N}$ und $x \in W$ geschrieben werden. Für $n = 0$ gilt $0x = 0_w$.

Bei einem K-Halbmodul W und einer Menge S_K mit $supp(S_K) = \{w_1, \ldots, w_n\}$ und $S(w_i) = k_i \in K, i = 1, \ldots, n$ kann eine Aggregation über diese Menge durchgeführt werden.

$$k_1 \star_w w_1 +_w \cdots +_w k_n \star_w w_n \in W$$

Eine Aggregation über eine leere Menge 0_w ergibt. Die Elemente von $K \times M$ werden mit $k \otimes m$ beschrieben und einfache Tensoren genannt.
Als Nächstes werden Multimengen von einfachen Tensoren betrachtet. Die Operation $+_{k \otimes M}$ stellt die Multimengenvereinigung und $0_{k \otimes M}$ die leere Multimenge da. Jede nichtleere Multimenge lässt sich dann durch einen Ausdruck $k_1 \otimes m_1 +_{K \otimes M} \cdots +_{K \otimes M} k_n \otimes m_n$ darstellen. Außerdem wird die Äquivalenz $k \star_{K \otimes M} \Sigma k_i \otimes m_i = \Sigma(k \cdot_K k_i) \otimes m_i$ definiert.

Sei $\sim$ die kleinste Kongruenz bezüglich $+_{K\otimes M}$ und $\star_{K\otimes M}$, dann gilt für alle k, k', m und m':

- $(k +_K k') \otimes m \sim k \otimes m +_{K\otimes M} k' \otimes m$
- $0_K \otimes m \sim 0_{K\otimes M}$
- $k \otimes (m +_M m') \sim k \otimes m +_{K\otimes M} k \otimes m'$
- $k \otimes 0_M \sim 0_{K\otimes M}$

```
SELECT AVG(grade)
FROM lecturers l, grades g
WHERE l.course_nr = g.course_nr
AND fullname = 'Profesor D';
```

Anfrage 8.3: Beispielanfrage für Provenance-Polynome

Beispiel Als Beispielanfrage für die Aggregation mit Provenance, wird die Fragestellung wie die Durchschnittsnote der Prüfungen bei PROFESSOR D ist gestellt. Die Fragestellung wird in 8.3 als SQL-Anfrage dargestellt.

course_nr	grade	Provenance
004	1.3	$L_4 \cdot G_{13}$
004	3.0	$L_4 \cdot G_{14}$
006	2.7	$L_6 \cdot G_{17}$
006	4.0	$L_6 \cdot G_{18}$

Tabelle 8.5: Ergebnis der Zwischenanfrage $\pi_{course\_nr,grade}(\sigma_{fullname='ProfesorD'}(Lecturers \bowtie Grades))$

Das Zwischenergebnis der Anfrage vor der Aggregation wird in der Tabelle 8.5 dargestellt. Durch die Selektion wird die Aggregation nur über vier Tupel durchgeführt, hierdurch wird der Provenance-Ausdruck am Ende übersichtlicher.

avg(grade)
2,75

Tabelle 8.6: Ergebnis der Beispielanfrage für Aggregation mit Provenanve.

Das Ergebnis der Anfrage ist in der Tabelle 8.6 dargestellt. Die ***how***-Provenance von dem Ergebnis ist:

$$\frac{1.3 \otimes L_4 \cdot G_{13} +_{K\otimes M} 3.0 \otimes L_4 \cdot G_{14} +_{K\otimes M} 2.7 \otimes L_6 \cdot G_{17} +_{K\otimes M} 4.0 \otimes L_6 \cdot G_{18}}{L_4 \cdot G_{13} +_{K\otimes M} L_4 \cdot G_{14} +_{K\otimes M} L_6 \cdot G_{17} +_{K\otimes M} L_6 \cdot G_{18}}$$

Dass die Operation $AVG()$ dem Ausdruck $\frac{SUM()}{COUNT()}$ gleicht, ist auch im Provenance-Polynom zu erkennen. Der Zähler des Ausdrucks entspricht der Provenance der SUM-Operation über den Werten. Im Nenner ist die Provenance der $COUNT$-Operation zu erkennen.

8.2.4. Weitere Aspekte der Data Provenance

Obwohl in diesem Kapitel der Schwerpunkt auf die Provenance-Halbringe gelegt wurde, konnten nicht alle Aspekte vollständig vorgestellt werden. Außerdem sind andere Bereiche der Data Provenance nicht ganz so genau betrachtet worden. Deswegen soll in diesem Abschnitt ein kleiner Überblick gegeben werden, welche weiteren Aspekte es in der Data Provenance gibt.

Gruppierung und geschachtelte Aggregation Auch wenn bereits der Schwerpunkt auf den Provenance-Halbringen lag, war es nicht möglich die in dem Artikel [ADT11] vorgestellte Provenance für Gruppierungen zu betrachten. Außerdem werden, zusätzlich zu den im Abschnitt 8.2.3 vorgestellten Aggregationen, in [ADT11] auch geschachtelte Aggregationen betrachtet.

Data Lineage Wie im Abschnitt 3.1.4 bereits beschrieben, existiert zusätzlich zu den Fragen ***how***, ***why***, ***why-not*** und ***where*** auch noch die ***lineage***. Diese ist eine der ersten Formen der Provenance und wird unter anderem in den Artikeln [WS97] und [CWW00] behandelt. Die Definitionen der ***lineage*** ist leider nicht fest und unterscheidet sich z. B. auch in den beiden Artikeln. In dem Provenance Tool **Trio** (Kapitel 9.3.1) wird eine weitere Art von ***lineage*** umgesetzt, welche der ***how***-Provenance ähnelt.

Provenance Games Eine Art für ***why-not***-Anfragen, eine Antwort zu finden, sind die Provenance-Games. Sie können aber auch auf in Ergebnis vorhandene Tupel angewendet werden, dann wird die ***how***-Provenance dieses Tupels durch einen Graphen dargestellt. Provenance-Games finden unter anderem Erwähnung in [HDB17] und werden in dem Tool **GProM** (Kapitel 9.3.4) umgesetzt.

9. Provenance-Tools

Provenance ist kein rein hypothetisches bzw. rein theoretisches Konstrukt, sondern wird tatsächlich angewendet. Hierfür gibt es verschiedene Software-Tools, welche die theoretischen Gedanken von Provenance auf Anwendungsfälle gezielt zuschneiden.

In diesem Abschnitt werden keine Provenance-Tools miteinander verglichen. Stattdessen werden die grundlegenden Messkriterien für Provenance-Tools erklärt und dann konkret auf ein Provenance-Tool eingegangen.

Es gibt auch Programme, welche den PROV-Standard des W3C umsetzen. Auf diese wird speziell im Abschnitt 10.2.3 eingegangen und deswegen werden sie in diesem Kapitel auch nicht behandelt. Eine Erklärung zu PROV findet sich im Unterkapitel 10.2.

9.1. Vielfalt an Provenance-Tools

Es gibt derzeit eine Reihe an Tools, also sowohl direkt benutzbaren Programmen als auch in andere Software integrierbare Softwarebibliotheken, mithilfe derer man Provenance anwenden kann. Auch sind diese Tools meist auf eine bestimmte Zielgruppe bzw. einen Anwendungsfall bzw. eine Kategorie von Anwendungsfällen zugeschnitten. Beispielsweise gibt es auf Wissenschaft zugeschnittene Tools und welche, die auf Datenanalyse, auf Geschäftslogik oder auf allgemeine Programmierung zugeschnitten sind. Weiterhin unterscheiden sich die vielen verschiedenen Tools in ihrer Granularität.

9.2. Granularität

Der Begriff *Granularität* (*Granularity*) im Kontext von Provenance-Tools beschreibt, wie detailliert die Ausgaben sind, die ein Tool generiert. Die Granularität ist oft davon abhängig, wie gut ein Workflow dokumentiert und für ein Provenance-Tool formuliert ist [HDB17, S. 893].

9.2.1. Grobkörnige Provenance

In den meisten Fällen ist im Allgemeinen nicht viel darüber bekannt, wie die einzelnen Resultate bzw. Items eines Workflows auf ihre Herkunft zurückzuführen sind. Bekannt sind nur, die Ein- und Ausgabeinformationen, jedoch nicht, wie diese auf der Ebene von beispielsweise einzelnen Zeilen zustande kommen. Man spricht dann von einem Black-Box-Ansatz In diesem Fall ist der Workflow als *coarse-grained*, also übersetzt *grobkörnig*, zu bezeichnen [HDB17, S. 893].

9.2.2. Feinkörnige Provenance

Bei einer *fine-grained*, also feinkörnigen, Provenance kann die Herkunft einzelner Daten nicht nur auf der Ebene des gesamten Prozesses bestimmt werden, sondern konkret für einzelne Items (z. B. Zeilen). Einige Tools verarbeiten hierzu diese Daten einzeln nacheinander. Dafür ist jedoch die Kenntnis über die internen Abläufe eines Workflows notwendig. Man spricht dann von einem White-Box-Ansatz. Einige Tools füttern den Workflow auch stückweise mit einzelnen Datensätzen, um diese Feinkörnigkeit zu erreichen [HDB17, S. 893].

9.3. Bisherige Untersuchungen

In einer Arbeit von 2021 wurden sieben Provenance-Tools untersucht und fünf davon anhand von Beispielen getestet [Fla+21]. Alle untersuchten Tools sind kostenlos und Open Source. Zur Untersuchung wurden vier Beispielanfragen verwendet, um die Unterstützung der Tools für einzelne Kriterien (z. B. Duplikatseliminierung) zu testen. Erklärt werden diese Anfragen im Einführungskapitel 2. Im Folgenden werden die Ergebnisse der Tool-Tests kurz zusammengefasst.

9.3.1. Trio

Das Tool *Trio* hat eine aufgeräumte Benutzeroberfläche namens *TrioExplorer*. Es fehlen jedoch einige Funktionen, die noch nicht implementiert wurden. Mit *TrioPlus* existiert auch ein Command-Line Interface (CLI) für Trio, jedoch gibt es hierfür keine Dokumentation. Trio kann drei der vier getesteten Beispielanfragen berechnen. Die Beispielanfrage zur Duplikatseliminierung wird von Trio nicht unterstützt. Außerdem funktionierte eine andere Anfrage nur unter Nutzung bestimmter Einstellungen.

Trio kann mit ***why***- und ***where***-Provenance umgehen, unterstützt aber keine ***how***-Provenance. Dazu kommt, dass Trio keine Unterstützung für die Befehle `DELETE` und `UPDATE` hat, obwohl diese in der Dokumentation von Trios eigener Abfragesprache *TrioQL* beschrieben sind [Fla+21].

9.3.2. ORCHESTRA

ORCHESTRA kann alle Beispielanfragen bis auf eine verarbeiten. Der Grund hierfür ist, dass ORCHESTRA grundsätzlich keine Aggregationen unterstützt, dementsprechend unterstützt es auch die Beispielanfrage mit Aggregationen nicht. ORCHESTRA unterstützt sowohl ***why***- als auch ***how***-Provenance. Bei der ***why***-Provenance bietet es Granularität auf Tupel-Ebene. Die verwendete Abfragesprache ist Datalog, daher wird Duplikatseliminierung von Grund auf unterstützt. ORCHESTRA brint eine eigene grafische Benutzeroberfläche mit und kann Provenance-Graphen für ***how***-Provenance anzeigen [Fla+21].

9.3.3. Perm

Das Tool *Perm* baut auf PostgreSQL auf und erweitert dabei die SQL-Syntax um ein `PROVENANCE`-Statement. Es bringt außerdem eine eigene grafische Benutzeroberfläche namens *Perm Browser* und einen Query-Rewriter zur Übersetzung von speziellen Provenance-Anfragen zu SQL-Anfragen mit. Perm unterstützt alle vier getesteten Beispielanfragen. Es kann die ***why***- und ***where***-Provenance berechnen, jedoch keine ***how***-Provenance [Fla+21].

9.3.4. GProM

GProM ist gut dokumentiert und einfach zu installieren. Es unterstützt verschiedene Datenbankanbindungen und Anfragesprachen (SQL und Datalog). *GProM* kann Provenance-Game-Graphen generieren und unterstützt ***why***- und ***why-not***-Provenance. Auch dieses Tool unterstützt alle vier Beispielanfragen, wobei eine umgeschrieben werden musste, da GProM das `JOIN`-Statement nicht unterstützt. GProM wird derzeit (Stand 2021) aktiv weiterentwickelt [Fla+21].

9.3.5. ProvSQL

Das Tool *ProvSQL* ist eine Erweiterung für PostgreSQL wird aktiv weiterentwickelt (Stand 2021). Es nutzt User-defined functions zur Berechnung der Provenance und unterstützt ***where***-, ***why***- und ***how***-Provenance sowie Provenance-Ringe. Von den vier Beispielanfragen unterstützt es eine nicht, da Aggregationen noch nicht unterstützt werden [Fla+21].

9.4. Kepler

Kepler[1] ist eine Anwendung, welche Wissenschaftler:innen, Analyst:innen und Softwareentwickler:innen dabei helfen soll, Modelle und Analysen zu erstellen und durchzuführen sowie diese mit einer breiten Menge anderer wissenschaftlicher und Ingenieurs-Disziplinen zu teilen [Kep].

Dabei kann Kepler auf Daten verschiedener Formate, auf die lokal oder über das Internet zugegriffen wird, operieren. Unterstützt wird beispielsweise auch das Kombinieren von kompiliertem C-Code mit R-Scripten. Mithilfe der grafischen Benutzeroberfläche können Anwender:innen sogenannte *scientific workflows*, also wissenschaftliche Abläufe, anlegen und die dazugehörigen Analysemodule und Datenquellen verbinden [Kep].

Kepler ist Java-basiert, also plattformunabhängig und funktioniert daher sowohl unter Windows als auch unter Linux und macOS. Zudem ist Kepler ein Open-Source-Projekt und steht unter der BSD-Lizenz zur Verfügung [Kep].

9.4.1. Scientific Workflow

Ein *Scientific Workflow* ist ein Werkzeug für den Zugriff auf und für die Durchführung komplexer Analysen auf wissenschaftlichen Daten. Ein solche Workflow besteht i. d. R. aus mehreren Schritten, die durch Aktoren repräsentiert werden [Cow15].

9.4.2. Aktoren

Aktoren („Actors“) können unterschiedliche Aktionen in Bezug auf Daten ausführen – zum Beispiel kann ein Aktor Daten aus einer Datenbank abfragen oder Daten auf dem Bildschirm ausgeben. Durch Verbindungen von Aktoren miteinander kann die Ausgabe eines Aktors (z. B. das Ergebnis einer Datenbankabfrage) als Eingabe für einen anderen Aktor genutzt werden. Dieser kann z. B. Daten filtern, verändern, aggregieren oder ausgeben.

Es ist auch möglich, einen Workflow oder einen Teil eines Workflows als Aktor zu verwenden. Diese *Composite Actor* genannte Methode erlaubt es, Konstrukte aus einem Workflow wiederzuverwenden [Cow15].

Über sogenannte Ports kann die Ein- bzw. Ausgabe eines Aktors mit dem jeweiligen Gegenstück eines anderen Aktors verbunden werden. Durch Doppelklick auf einen Aktor können die Parameter für diesen eingestellt werden. Abb. 9.1 zeigt das Fenster zur Konfiguration der Parameter eines `Directory Listing`-Aktors.

[1] `https://kepler-project.org/index.html`

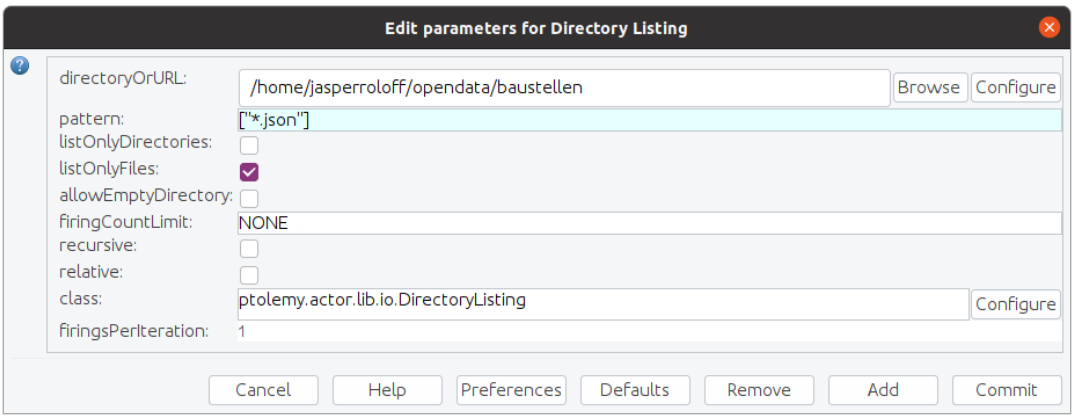

Abbildung 9.1: Parameter für einen Aktor

9.4.3. Direktoren

Direktoren („Directors“) beeinflussen, wie ein Workflow ausgeführt wird. Jeder Workflow muss zwingend einen Direktor haben. Mithilfe von Direktoren kann konfiguriert werden, ob ein Workflow einmalig durchläuft oder kontinuierlich bis zum manuellen Stopp wiederholt wird. Es gibt auch Direktoren, welche asynchrone Ausführung von Workflows ermöglichen. Je nach gewünschter Struktur und abhängig von den eingesetzten Aktoren kann es nötig sein, einen bestimmten Direktor zu verwenden [Cow15].

Im später gezeigten Beispiel kommt der `SDF Director` zum Einsatz.

9.4.4. Benutzeroberfläche von Kepler

In Kepler können Aktoren per Drag-and-Drop angeordnet und miteinander verbunden werden. Die grafische Benutzeroberfläche von Kepler besteht neben dem Bereich zum Erstellen und Bearbeiten eines Workflows auch aus einer Toolbar zur Steuerung des Programms und aus einem *Components-, Data Access- & Outline*-Bereich. Unter *Components* finden sich die verfügbaren Aktoren, die dann per Drag-and-Drop in den Arbeitsbereich kopiert werden können. Außerdem gibt es darunter eine Miniatur-Ansicht des Workflows.

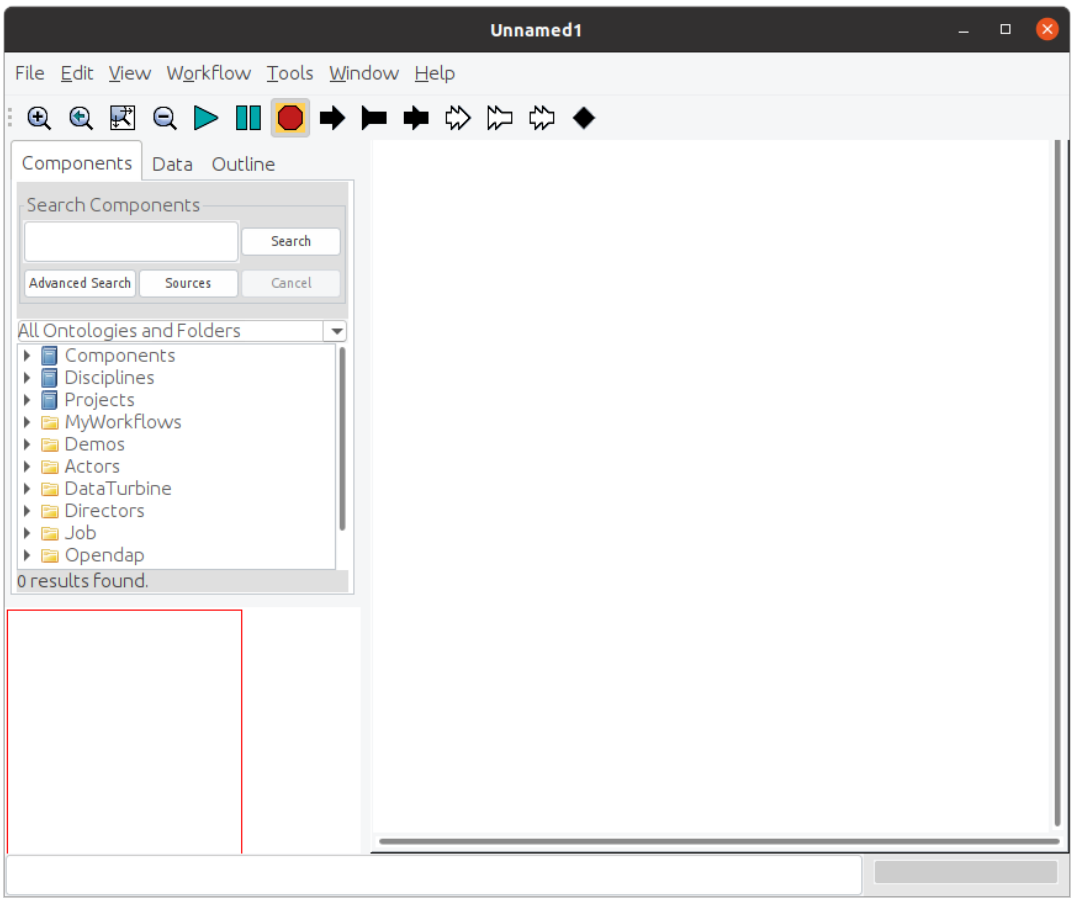

Abbildung 9.2: Benutzeroberfläche von Kepler

9.4.5. Beispiel

Zur Demonstration wurde in Kepler ein Workflow angelegt, welcher aus einem Verzeichnis mit JSON-Dateien den Inhalt der ersten Datei ausliest, verarbeitet und für jedes Element aus einem Array in einem bestimmten Attribut ein bestimmtes Attribut ausgibt.

Konkret ist das Beispiel ein Verzeichnis mit JSON-Dateien über Baustellen in der Hanse- und Universität Rostock, welche von *OpenData.HRO*, dem Open-Data-Portal der Stadt, heruntergeladen wurden.

Jede dieser Dateien enthält ein JSON-Objekt, welches ein Attribut namens `features` hat. Dieses Attribut ist vom Typ Array, sodass es eine Liste mit Unterobjekten enthält. Diese Unterobjekte haben jeweils ein Attribut vom Typ Objekt namens `properties`, welches unter anderem ein Attribut namens `sparte` hat.

Die Struktur der JSON-Dateien wird in folgendem Listing dargestellt:

```
1 {
2   "type": "FeatureCollection",
3   "features": [
4     {
5       "type": "Feature",
6       "properties": {
7         "uuid": "c0e668be-c53b-11e8-b368-0050569946ac",
8         "kreis_name": "Rostock",
```

```
        "kreis_schluessel": "13003",
        "gemeindeverband_name": "Rostock, Hanse- und Universit\
           u00e4tsstadt",
        "gemeindeverband_schluessel": "130030000",
        "gemeinde_name": "Rostock, Hanse- und Universit\u00e4tsstadt",
        "gemeinde_schluessel": "130030000000",
        "gemeindeteil_name": "Brinckmansdorf",
        "gemeindeteil_schluessel": "0020",
        "strasse_name": "Dierkower Damm",
        "strasse_schluessel": "01820",
        "lage": "zwischen An der Zingelwiese und Hausnummer 47",
        "sparte": "Wasserleitung",
        "verkehrsbeeintraechtigung_art": "halbseitige Sperrung Fahrbahn"
           ,
        "verkehrsbeeintraechtigung_beschreibung": "halbseitige Sperrung
           des Verkehrs; Sicherungsma\u00dfnahmen entlang der Stra\
           u00dfe; Sicherungsma\u00dfnahmen entlang des Gehwegs;
           Sperrung des Fu\u00dfg\u00e4ngerverkehrs im Gehwegbereich",
        "grund": "Sanierung und Neubau Trinkwasserleitung",
        "verkehrsbeeintraechtigung_anfang": "2017/11/13 00:00:00+01",
        "verkehrsbeeintraechtigung_ende": "2019/06/29 00:00:00+02"
      },
      "geometry": {
        "type": "Point",
        "coordinates": [
          12.15724047225227,
          54.09776401167255
        ]
      }
    },
    ...
  ]
}
```


Wie im Bildschirmfoto (Abb. 9.3) zu sehen, nutzt der Workflow den `SDF Director`. Außerdem wurde abseits des eigentlichen Workflows ein `ShowTypes`-Aktor platziert. Dieser sorgt dafür, dass die Benutzeroberfläche von Kepler die Typdefinitionen für die Ein- und Ausgabeports von Aktoren anzeigt – dies ist für das Debugging sehr hilfreich.

Der eigentliche Workflow besteht aus folgenden Aktoren:

Directory Listing Dieser Aktor listet den Inhalt eines vorgegebenen Verzeichnisses auf und gibt ihn als Liste von Strings weiter. Dabei sind Pfad zum Verzeichnis und ein Muster zum filtern nach Dateinamen konfigurierbar.

Array Element Der Aktor `Array Element` nimmt einen Array als Eingabe entgegen und gibt das Element an einer bestimmten Position im Array als Ausgabe weiter. Die Position des Elements kann konfiguriert werden. Im Beispiel ist sie `0`. Da die Eingabe vom Typ

`arrayType(string)` ist, ist die Ausgabe vom Typ `string`. In diesem Fall ist der Inhalt dieser Zeichenkette der Pfad zur Datei.

Simple File Reader Der `Simple File Reader` nimmt einen Dateipfad (String) als Eingabe entgegen und gibt den Inhalt der jeweiligen Datei als String wieder aus.

JSONToToken Dieser Aktor nimmt einen JSON-String entgegen und gibt ein Token aus, welches die durch den JSON-String beschriebenen Daten repräsentiert. Da vor Ausführung unbekannt ist, von welchem Typ das Token oder dessen Attribute sind, muss eine Typdefinition vorgenommen werden.

Expression Ein `Expression`-Aktor kann u. a. einfache mathematische Ausdrücke verarbeiten. In diesem Fall wird ein Attribut von einem Objekt abgefragt. Die Formel kann durch Doppelklick auf den Aktor konfiguriert werden.

Array To Sequence Dieser Aktor nimmt einen Array als Eingabe entgegen und gibt danach die Elemente des Arrays einzeln nacheinander aus, sodass sie jeweils die nachfolgenden Aktoren durchlaufen.

Expression2 Dies ist wieder ein `Expression`-Aktor, welcher in diesem Fall wieder ein Attribut eines Objekts abfragt und ausgibt.

Monitor Value Dies ist ein Aktor, welcher die Eingabe direkt anzeigt. Im Bild ist zu sehen, dass die dortige Eingabe „Wasserleitung“ ist. Neben `Monitor Value` gibt es auch andere Aktoren, welche Ausgaben auf dem Bildschirm anzeigen können (z. B. als separates Fenster).

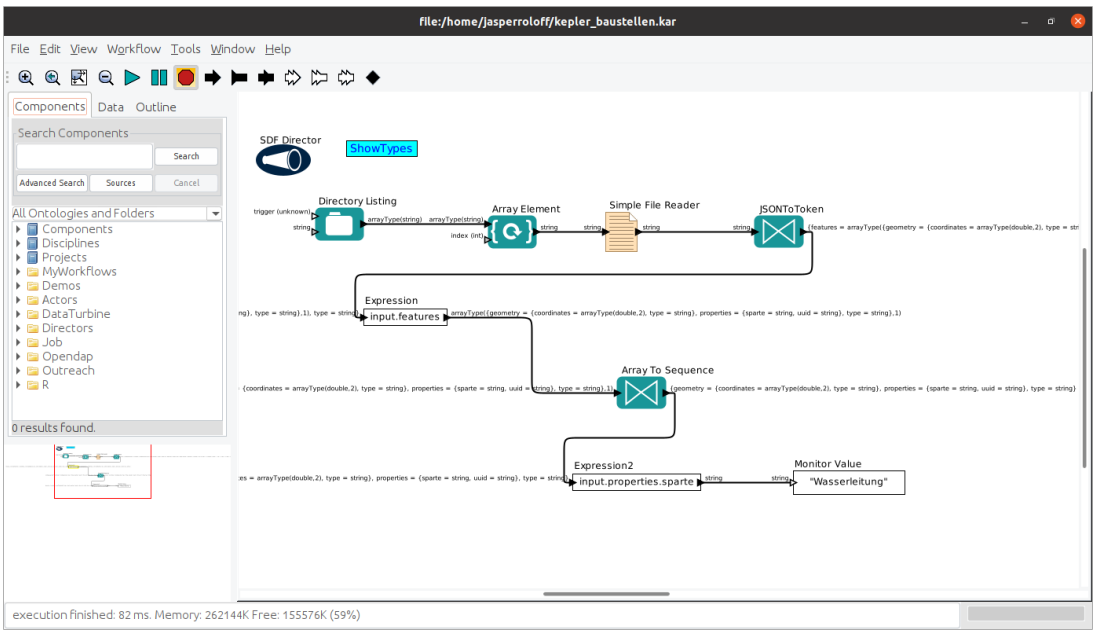

Abbildung 9.3: Beispiel-Workflow

9.4.6. Besonderheiten

Während der Erstellung des Beispiel-Workflows in Kepler wurden einige Besonderheiten festgestellt, die für die Benutzung von Kepler sehr wichtig sind, aber nicht besonders gut bzw. nicht an der richtigen Stelle in der Dokumentation erwähnt sind.

- Der `JSONToToken`-Aktor kann Strings als JSON-Objekt einlesen und als Objekt (in Kepler sog. Token) ausgeben. Die Ausgabe ist dabei standardmäßig vom Typ `unknown`. Da Kepler eine Typprüfung vornimmt und andere Aktoren diesen Typ nicht als Eingabe akzeptieren, schlägt die Ausführung des Workflows zunächst einmal fehl. Um dieses Problem zu lösen und die Ausgabe des `JSONToToken`-Aktors weiter verarbeiten zu können, ist es nötig, auf dem Ausgabeport des Aktors eine Typdefinition vorzunehmen. Aufgrund der Typprüfung von Kepler werden die Typen für Ein- und Ausgaben bei weiteren Aktoren im Normalfall automatisch erkannt. Mittels eines Rechtsklicks auf den Port und Auswahl der Option „Configure Port“ im Kontextmenü (Abb. 9.4) ist es möglich, die Typdefinition vorzunehmen (siehe Abb. 9.5).

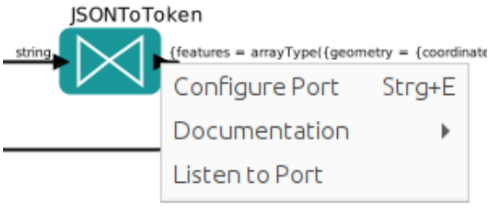

Abbildung 9.4: Kontextmenü eines Ports

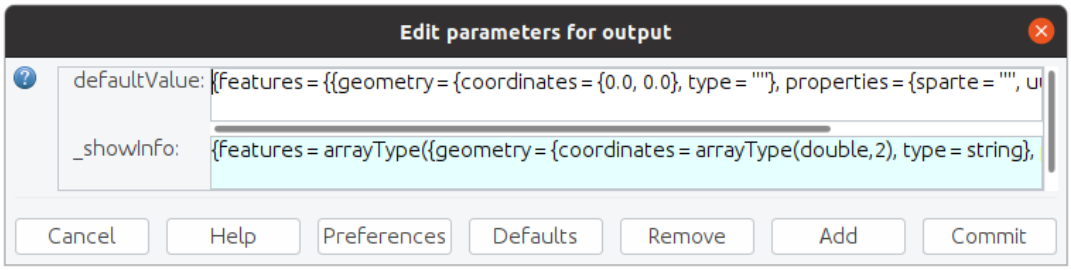

Abbildung 9.5: Typdefinition für Port

- Mithilfe des `Expression`-Aktors können gängige Ausdrücke genutzt werden, um z. B. arithmetische Operationen vorzunehmen oder Attribute von Objekten abzufragen. In der initialen Variante hat ein neu erstellter `Expression`-Aktor jedoch keine Eingabeports, sondern nur einen Ausgabeport. In Kepler können die Ports über das Kontextmenü (Abb. 9.6) bearbeitet werden. Dabei ist das Hinzufügen weiterer Ports sowie das Umbenennen von Ports möglich (siehe Abb. 9.7).

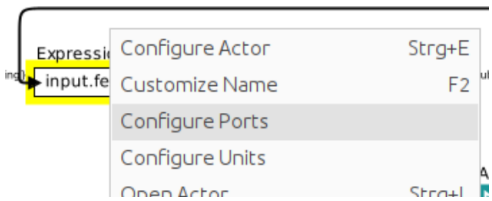

Abbildung 9.6: Kontextmenü eines Aktors

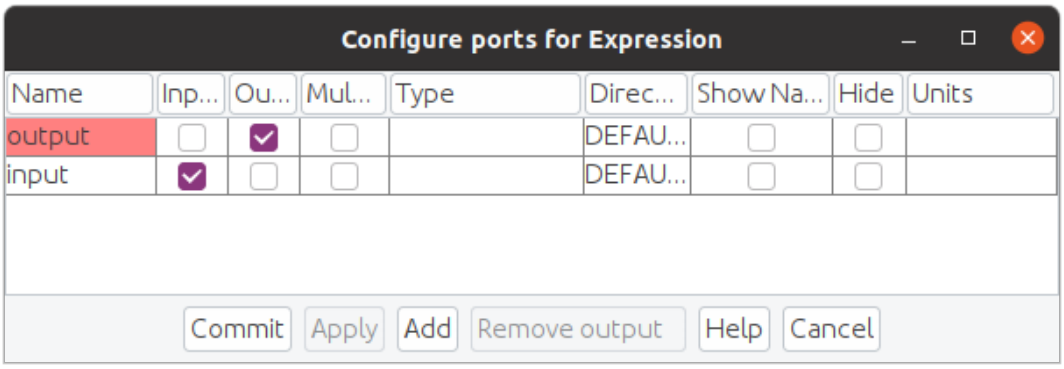

Abbildung 9.7: Portkonfiguration für Aktor

10. Provenance im Web

Das Internet hat sich in den letzten Jahrzehnten zu einer großen Plattform entwickelt, welches den einfachen Austausch von Wissen ermöglicht. Vor allem für Wissenschaft und Wirtschaft ist es wichtig, dass Entstehungsprozesse von Dokumenten nachvollziehbar sind. Allerdings wird dies durch die rasante Entwicklung des Web erschwert. In diesem Kapitel erfolgt zuerst eine Aufzählung der darunterliegenden Probleme. Dabei wird darauf eingegangen, wie sich die Provenance einordnet. Danach wird PROV, ein W3C-Standard, als konkreter Lösungsvorschlag vorgestellt. Abschließend folgt eine Auflistung einiger Tools, welche den PROV-Standard unterstützen.

10.1. Allgemeine Herausforderungen

Um die Provenance als notwendigen Bestandteil im Datenmanagement zu verstehen, wollen wir sie in dessen Anforderungsprofil einsortieren. Verbildlicht werden die einzelnen Anforderungen und Beispiele für Standards in der Abbildung 10.1. In die einzelnen Standards soll an dieser Stelle nicht tiefgründig eingegangen werden.

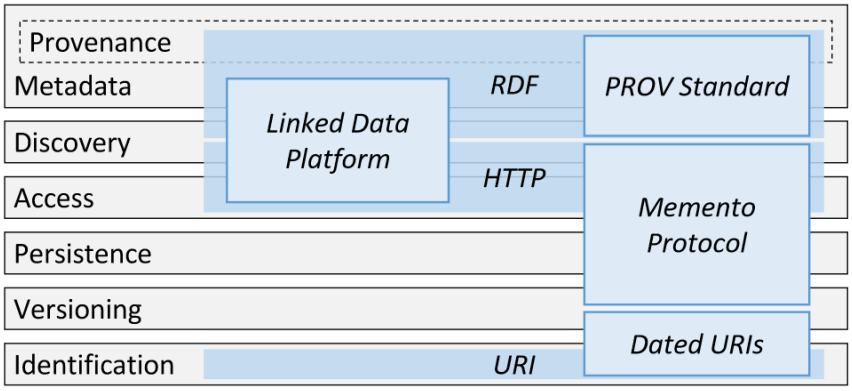

Abbildung 10.1: Anforderungen an das Datenmanagement im Web und dazugehörige Standards [Gle+21]

Gleim et al. fassen in [Gle+21] die folgenden Anforderungen an das Datenmanagement im Web zusammen. Zunächst muss eine eindeutige Identifikation der Dokumente erfolgen. Dazu wird ein *Uniform Resource Identifier*, kurz *URI*, eingesetzt. Sie ist das Äquivalent zu der DOI für wissenschaftliche Dokumente oder der ISBN für Bücher. Weil sich Ressourcen im Internet verändern können, müssen unterschiedliche Versionen desselben Objektes referenzierbar sein. URIs können hierzu mit Versionsnummern oder Zeitstempeln versehen werden. Letzteres wird vom *World Wide Web Consortium* (W3C) für separate Dokumente empfohlen [The06]. Die *Web Accessibility Initiative* (WAI) des W3C legt für eigene Standards fest, dass für konkrete Versionen URIs mit Zeitstempeln verwendet werden, während ein Verweis auf die aktuellste Version kein Datum beinhaltet [Hen18]. Als nächster Schritt wird die Persistenz von Dokumenten genannt, welches der Langzeitarchivierung dienen soll. Nun wird ein Zugangsmechanismus (Access) benötigt. Zu diesem Zweck wird HTTP genutzt. Dokumente haben nur dann einen wirklichen Nutzen, wenn sie auch auffindbar sind (Discovery). Auch diese Aufgabe ist durch die Natur des HTTP abgedeckt. Das Memento Protocol kann ebenfalls für diese Aufgabe benutzt werden und deckt die Versionierung, die Persistenz, den Zugang und die Auffindbarkeit ab. Zuletzt wer-

den Metadaten benötigt, um die Daten entsprechend zu beschreiben. Eingesetzt werden hierzu die *Linked Data Platform* (LDP) im Zusammenhang mit dem *Resource Description Framework* (RDF). Die Provenance wird in [Gle+21] als Teil der Metadaten angesehen. Genau hier setzt der PROV-Standard an, welcher im folgenden Abschnitt behandelt wird.

10.2. W3C PROV

Das W3C hat sich des Problems angenommen. Es entwickelte mit PROV [GM13b] ein Datenformat zum standardisierten Austausch von Provenance-Informationen. Die dazugehörigen Dokumente befinden sich offiziell noch in Arbeit, aber wurden seit April 2013 nicht geändert. Vom W3C wird das Ziel von PROV folgendermaßen beschrieben:

> „The goal of PROV is to enable the wide publication and interchange of provenance on the Web and other information systems.“ [GM13b]

Man erkennt, dass das Internet in der Rolle als Informationssystem betrachtet wird. Dies spricht für die Information System Provenance. Sie wurde in Kapitel 6 ausführlicher behandelt. Ein Überblick über die einzelnen Bestandteile des Standards bietet die Abbildung 10.2. Die Grundlage bildet die Definition des Datenmodells (PROV-DM), wobei weitere Spielregelen in einem separaten Dokument beigelegt werden (PROV-CONSTRAINTS). Das Primer ist das offizielle Tutorial des Standards. Zusätzlich gibt es drei Serialisierungen: PROV-O, PROV-XML und PROV-N. Die restlichen Bestandteile sind hier nicht relevant.

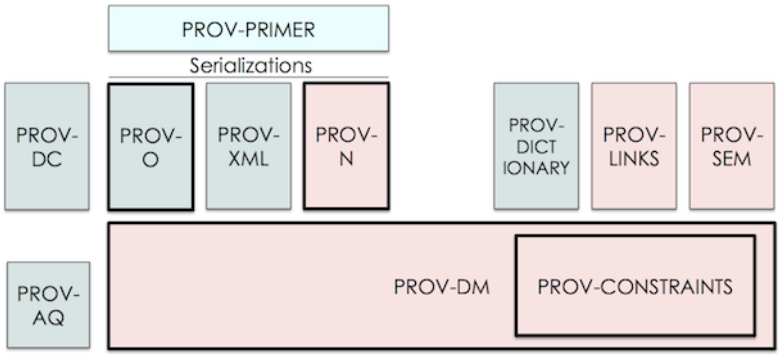

Abbildung 10.2: Überblick über PROV [GM13b]

10.2.1. Grundlegende Elemente von PROV

In PROV gibt es drei Kernelemente: Entitäten, Aktivitäten und Agenten.

Definition 10.1 (Entität, übernommen und übersetzt aus [MM13a])**.**
Eine **Entität** ist eine physische, digitale, konzeptionelle Art einer Sache mit festen Aspekten; Entitäten können real oder imaginär sein.

Entitäten sind alle Dinge, über welche eine Aussage zur Herkunft getroffen werden kann. In unserem Beispiel kann dies ein Datensatz der Studententabelle sein oder auch das Prüfungsprotokoll. Besonders auffällig an dieser Definition ist, dass nicht real existierende Entitäten erlaubt

sind. Beispielsweise kann in der Modellierung eine Entität hinzugefügt werden, welche bisher nur gedanklich existiert.

Definition 10.2 (Aktivität, übernommen und übersetzt aus [MM13a])**.**
Eine **Aktivität** ist etwas, das über einen Zeitraum auftritt und auf oder mit *Entitäten* arbeitet; sie darf Entitäten konsumieren, verarbeiten, umwandeln, verändern, verschieben, nutzen oder generieren.

Aktivitäten sind Prozesse. Sie sind in der Lage, Entitäten zu nutzen oder zu erschaffen. Beispiele sind die Immatrikulation, welche einen neuen Datensatz in der Tabelle erzeugt, oder das Erstellen des Prüfungsprotokolls aus den Informationen der Anmeldung.

Definition 10.3 (Agent, übernommen und übersetzt aus [MM13a])**.**
Ein **Agent** ist etwas, das eine gewisse Verantwortung über eine stattfindende *Aktivität*, die Existenz einer *Entität* oder die *Aktivität* eines anderen Agenten trägt.

Agenten tragen eine Verantwortung über Entitäten oder Aktivitäten. Die Universität selbst kann als Agent auftreten, weil sie die Informationen der Studierenden verarbeiten. PROV unterscheidet dabei *Personen*, *Organisationen* und *Softwareagenten*, welche in der Ontologie als Unterklassen angesehen werden [LSM13]. Dadurch sollen die meisten Anwendungsfälle abgedeckt werden.

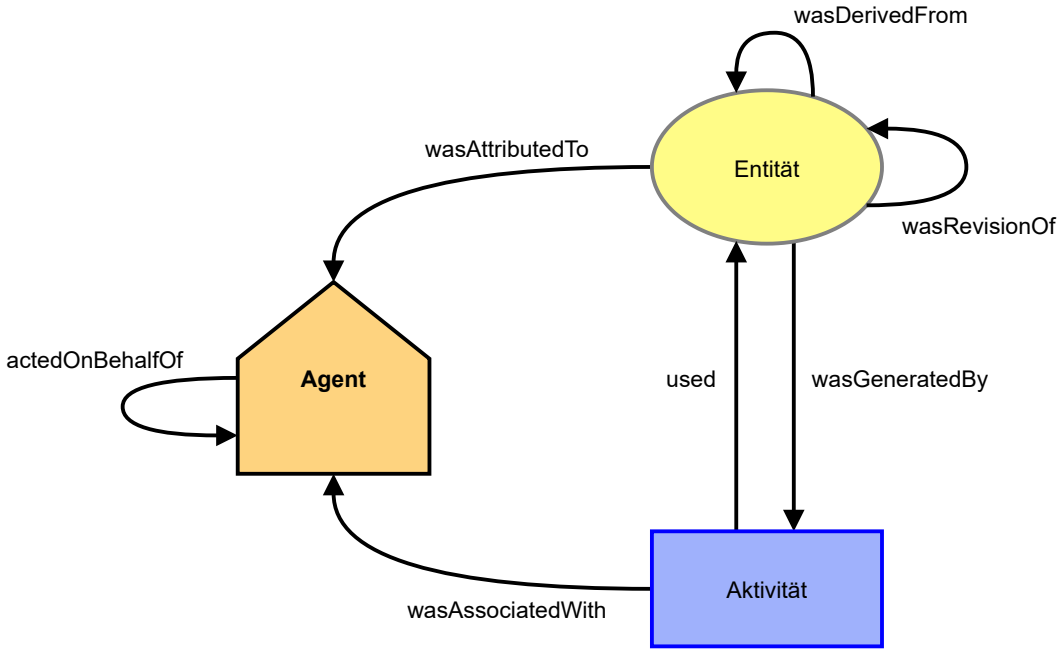

Abbildung 10.3: Elemente von PROV und erlaubte Verbindungen, nach [GM13a]

Wie die Elemente in PROV verbunden werden dürfen, zeigt die Abbildung 10.3. In der grafischen Repräsentation werden Entitäten als gelbe Ovale mit grauen Rahmen gezeichnet. Aktivitäten sind blaue Rechtecke und Agenten werden in orangefarbene „Häuschen“ platziert. Die Relationen haben folgende Bedeutungen:

Definition 10.4 (Beziehungen in PROV, übernommen und übersetzt aus [MM13a])**.**

wasDerivedFrom: Eine **Ableitung** (**Derivation**) ist eine Transformation einer Entität in eine andere, eine in eine neue Entität resultierende Aktualisierung oder die Konstruktion einer Entität auf Basis einer vorher existenten Entität.

wasRevisionOf: Eine **Revision** ist eine Ableitung, bei welcher die resultierende Entität eine überarbeitete Version eines Originals ist.

used: Eine **Verwendung** (**Usage**) ist der Anfang der Nutzung einer Entität durch eine Aktivität. Vor der Verwendung begann die Aktivität nicht, diese Entität zu benutzen, und konnte auch nicht durch diese beeinflusst werden.

wasGeneratedBy: **Generation** ist der Abschluss einer Produktion einer neuen Entität durch eine Aktivität. Die Entität existierte nicht vor der Generation und ist nach der Generation verfügbar für die Verwendung.

wasAttributedTo: **Zuschreibung** (**Attribution**) ist die Zurechnung einer Entität zu einem Agenten.

wasAssociatedWith: Eine Aktivitäts-**Assoziation** ist die Verantwortungszuweisung an einen Agenten für eine Aktivität, dies deutet darauf hin, dass der Agent eine Rolle in dieser Aktivität besaß. Weiterhin erlaubt es eine Spezifizierung eines Plans, mittels dessen der Agent ein bestimmtes Ziel im Kontext dieser Aktivität anstrebt.

actedOnBehalfOf: **Delegation** ist die Zuweisung der Autorität und Verantwortung zu einem Agenten, eine spezifische Aktivität als Vertreter durchzuführen, während der Agent, in dessen Auftrag er arbeitet, etwas Verantwortung für das Ergebnis der Arbeit behält.

Es gibt weitere Beziehungen, auf welche hier nicht weiter eingegangen werden soll. Eine vollständige Auflistung kann dem Standard entnommen werden, siehe dazu [MM13a].

Beispiel Im Unterabschnitt 7.4 bzw. in der Abbildung 7.2 haben wir bereits den Workflow einer Prüfungsanmeldung in BPMN modelliert. Dabei musste eine eingehende Prüfungsanmeldung formal geprüft werden. Daraufhin wurde ein Prüfungsprotokoll erstellt und dem Dozenten übergeben. Nehmen wir an, der Prüfungsausschuss der Universität möchte die Abläufe des Prüfungsamtes überprüfen. Dabei soll ein stärkerer Blick auf die Dokumente und deren Entwicklung gelegt werden als auf den Workflow als Ganzes. Mit Hilfe der oben besprochenen Elemente ist es uns möglich, diesen Sachverhalt in PROV darzustellen.

Betrachten wir zunächst die Entitäten und Aktivitäten. Das Diagramm beginnt mit dem Studenten-Datensatz und dem Formular, welche Entitäten darstellen. Beide werden in einer Aktivität *prüfen1* verwendet. Also ziehen wir von der Aktivität aus je einen Pfeil zu den beiden Entitäten und beschriften diese mit *used*. Das Formular wurde ordnungsgemäß ausgefüllt, sodass das Prüfungsamt dieses gegenzeichnen kann. Weiterhin fügt es Informationen über den Studierenden hinzu, welche dieser nicht explizit aufschreiben musste, beispielsweise das Fachsemester. Dadurch wird *Bestätigtes-Formular* durch die Aktivität generiert. So entsteht der mit *wasGeneratedBy* beschriftete Pfeil von *Bestätigtes-Formular* zu *prüfen1*. Nun wird geprüft, ob eine

Registrierung zur Prüfung tatsächlich erfolgen darf. Dazu werden die *Zugangsvoraussetzungen* herangezogen. Ein *Anmeldung-Datensatz* wird durch *prüfen2* erstellt. In der Datenbank ist die Anmeldung nun existent, es fehlt nur noch ein *Prüfungsprotokoll*. Auf analoge Art und Weise stellt die Aktivität *erstellen* das Protokoll aus dem frisch erzeugten Datensatz zusammen.

Es verbleiben die Agenten, welche jetzt besprochen werden. *Herr Müller*, welcher als Person auftritt, ist für die formale Überprüfung der Anmeldungen zuständig. Die Aktivität *prüfen1* ist mit ihm assoziiert, in PROV wird diese Relation als *wasAssociatedWith* dargestellt. Er handelt hier nicht als Privatperson, sondern im Namen des Prüfungsamtes. Um diesen Sachverhalt zu verdeutlichen, nehmen wir das *Prüfungsamt* als Agenten des Typs Organisation mit auf und verbinden beide Agenten mit der Relation *actedOnBehalfOf*. Der Datensatz für die Prüfungsanmeldung und das Prüfungsprotokoll erstellt Herr Müller nicht selbst, sondern verwendet eine *Prüfungssoftware* für diese Aufgabe. Hier trägt also ein Softwareagent Verantwortung über die verbundenen Entitäten und Aktivitäten.

Das Ergebnis ist in Abbildung 10.4 zu sehen. Auffällig ist, dass die Pfeile zwischen Entitäten und Aktivitäten dem zeitlichen Verlauf entgegengesetzt sind. Begründet ist dies dadurch, dass bei der Provenance auf bestehende Informationen zurückgegriffen wird, um neue Dokumente zu erzeugen.

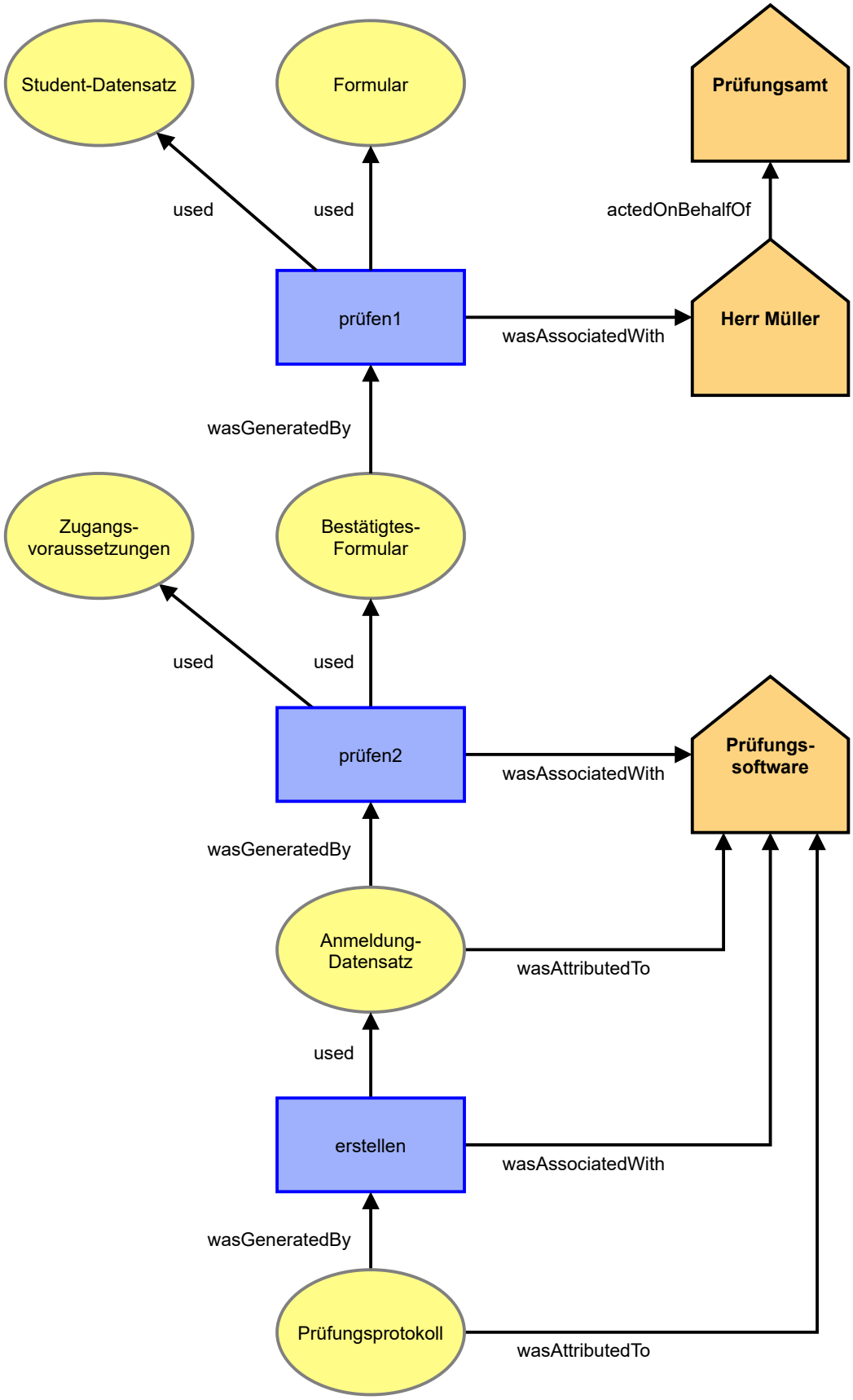

Abbildung 10.4: Modellierung einer Prüfungsanmeldung mit PROV

10.2.2. Serialisierungen zum Austausch

Die oben gezeigte Visualisierung ist für den Menschen gut zugänglich, allerdings kein geeignetes Datenformat für den Computer. Daher beschäftigt sich dieser Teil mit der Serialisierung von PROV. Genauer wird auf PROV-N eingegangen, währenddessen werden weitere Konzepte vorgestellt. Bevor diese jedoch gezeigt werden, soll das Konzept des Namensraums kurz erklärt werden, welche hierbei eine Rolle spielt.

Mit Hilfe von Namensräumen können Elemente gruppiert und so beispielsweise bestimmten Organisationen oder Webseiten zugeordnet werden. Auch für die Definition anwendungsspezifischer Eigenschaften können sie genutzt werden. Sie werden am Anfang eines PROV-Dokumentes definiert. Jeweils wird ein Präfix zur eindeutigen Identifizierung zugeordnet. Innerhalb des Dokumentes selbst werden diese dann verwendet. Der zu PROV selbst gehörende Namensraum ist `http://www.w3.org/ns/prov#` und erhält innerhalb des Standards den Präfix `prov` [GM13b]. Beliebig viele eigene Namensräume können eingeführt werden.

PROV-N Ausführlicher vorgestellt wird hier PROV-N. Diese Notation wurde für leichte Lesbarkeit entwickelt [MM13b]. Die Syntax ähnelt der Programmiersprache Prolog. Beginnen wir mit dem Präfix. Unsere Beispieluniversität soll die Domain `https://example.org/` besitzen. Definiert wird ein Namensraum, welcher für alle Elemente des Beispiels eingesetzt werden soll. In der Syntax wird zuerst das Schlüsselwort `prefix` geschrieben, dann das gewünschte Präfix selbst und zuletzt die URI in `<...>`:

```
prefix ex <https://example.org/>
```

Entitäten werden mit dem Schlüsselwort `entity` eingeführt. Alle Attribute werden in Klammern platziert:

```
entity(ex:Anmeldung-Datensatz)
```

Aktivitäten werden auf dieselbe Art und Weise notiert, das Schlüsselwort lautet `activity`:

```
activity(ex:erstellen)
```

Sie können um Zeitangaben ergänzt werden und somit einen Startzeitpunkt und ein Endzeitpunkt haben. Nehmen wir an, die Aktivität *erstellen* begann am 15. Juli 2021 um 12:34 Uhr und dauerte drei Sekunden, dann kann die Aktivität so umgeschrieben werden:

```
activity(ex:erstellen, 2021-07-15T12:34:56, 2021-07-15T12:34:59)
```

Agenten werden analog gelistet:

```
agent(ex:Herr-Müller)
agent(ex:Prüfungsamt)
agent(ex:Prüfungssoftware)
```

Wie oben eingeführt, gibt es drei Untertypen von Agenten: Person, Organisation und Softwareagent. In PROV-N werden sie als optionale Attribute hinzugefügt. Die Eigenschaft hat die

Bezeichnung `prov:type`. Wenn man diese den bestehenden Agenten hinzufügt, sieht das folgendermaßen aus:

```
agent(ex:Herr-Müller, [ prov:type='prov:Person' ])
agent(ex:Prüfungsamt, [ prov:type='prov:Organization' ])
agent(ex:Prüfungssoftware, [ prov:type='prov:SoftwareAgent' ])
```

Nun folgen die Beziehungen zwischen den einzelnen Entitäten, Aktivitäten und Agenten. Die erste Beziehung ist `used`. Sie besteht beispielsweise zwischen dem Erstellungsprozess und dem Datensatz der Anmeldung. Die Notation hat drei Elemente:

```
used(ex:erstellen, ex:Anmeldung-Datensatz, -)
```

Hier fällt auf, dass als drittes „Argument“ ein `-` angegeben ist. An dieser Stelle gehärt ein Zeitstempel, welcher aber ausgelassen werden kann. Genauso verhalten sich auch `wasGeneratedBy`, `wasAssociatedWith` und `actedOnBehalfOf`:

```
wasGeneratedBy(ex:Prüfungsprotokoll, ex:erstellen, -)
wasAssociatedWith(ex:erstellen, ex:Prüfungssoftware, -)
actedOnBehalfOf(ex:Herr-Müller, ex:Prüfungsamt, -)
```

Zuletzt sei noch `wasAttributedTo` zu erwähnen. Es trägt kein Zeitstempel wie die vorher genannten Relationen.

```
wasAttributedTo(ex:Prüfungsprotokoll, ex:Prüfungssoftware)
```

Alle Beziehungen können optionale Attribute besitzen, wie bei unseren Agenten vorgestellt. Das vollständige PROV-N-Beispiel befindet sich im Anhang auf Seite 70.

10.2.3. Tools zur Arbeit mit PROV

Der PROV-Standard wurde innerhalb der letzten Jahre in mehreren Tools umgesetzt. Wert wird daher auf die Unterstützung des PROV-Standards gelegt. Hier gelistete Programme sind nicht mit jenen im Kapitel 9 zu verwechseln. Sie beschäftigen sich nicht mit dem W3C-Standard, sondern allgemein mit dem Bereich Provenance.

Git2PROV Git2PROV [Nie+13] stammt aus dem Jahr 2013 und ist somit eine der ersten Umsetzungen. Die Entwickler verfolgten die Idee, Git-Historien mit PROV zu modellieren. Implementiert ist es als node.js-Anwendung und unterstützt mehrere Serialisierungen. Der Quellcode ist GPLv3-lizenziert und auf GitHub[2] verfügbar. Zusätzlich kann es aus dem npm-Repository[3] geladen werden. Die Abbildung 10.5 zeigt, wie Nies et al. die Übersetzung von Git-Commits auf PROV-Elemente vorsehen. Jeder Commit c ist eine eigene Aktivität. Ein solcher ist in der Mitte des Bildes dargestellt. Autor und Committer sind in ihren Rollen als Agenten mit c assoziiert, im Bild links dargestellt. Dateien können durch Commits erstellt, geändert oder gelöscht werden. Dargestellt werden sie durch Entitäten f_c und f_{c-1}, welche die neue bzw. alte Version

[2] `https://github.com/IDLabResearch/Git2PROV`
[3] `https://www.npmjs.com/package/git2prov`

repräsentieren. Eine allgemeine Sicht auf die Datei wird durch die Entität *f* realisiert. Die Relation *specializationOf* verbindet sie mit ihren Versionen. In der Grafik sind alle Entitäten auf der rechten Seite eingezeichnet.

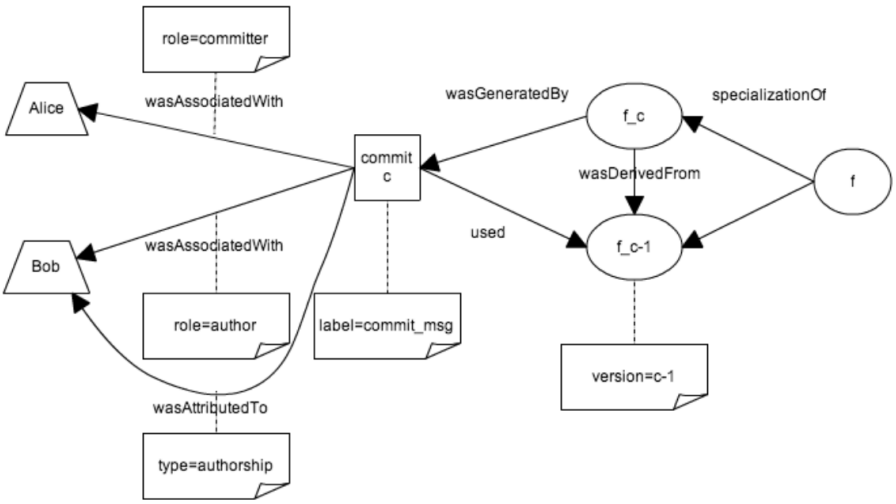

Abbildung 10.5: Modellierung von Git-Commits in Git2PROV [Nie+13]

PROV-O-Viz PROV-O-Viz [HG14] wurde 2014 veröffentlicht. Es greift auf Sankey-Diagramme zurück, welche in Abschnitt 4.1 erwähnt wurden. Implementiert ist es in Python 2. Hoekstra und Groth verfolgen dabei zwei Ziele, hier übersetzt:

> „1. wichtige Aktivitäten auf Grundlage des Datenflusses feststellen; und
> 2. verstehen, wie Daten durch eine ausgewählte Aktivität fließen.“ [HG14, Seite 216]

Um das erste Ziel zu erreichen, werden informationsreiche Aktivitäten größer dargestellt als andere. Für die zweite Zielsetzung wurde ein Dropdown-Menü eingefügt, welches die Auswahl einer bestimmten Aktivität ermöglicht. Sie wird dann angezeigt. Die GUI kann der Abbildung 10.6 entnommen werden, welches ebenfalls aus dem Paper stammt. Wie der Name vermuten lässt, nimmt es Eingaben in PROV-O entgegen. Das Tool steht unter der MIT-Lizenz und ist quelloffen. Der Code kann auf GitHub[4] eingesehen werden.

[4] `https://github.com/Data2Semantics/provoviz`

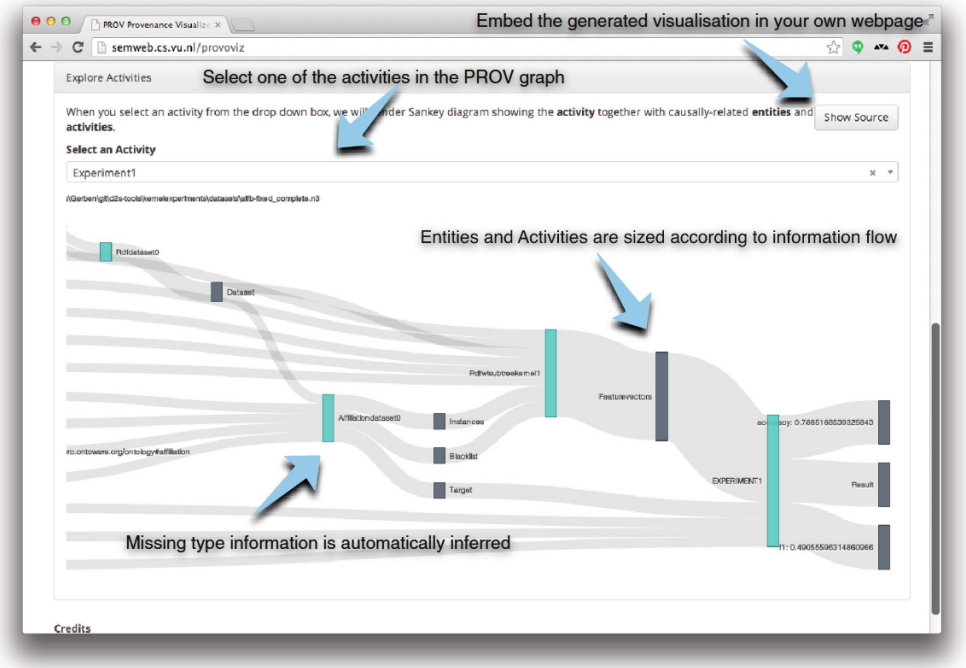

Abbildung 10.6: Visualisierung in PROV-O-Viz [HG14]

ProvViz Auf der ProvenanceWeek 2021[5] wurde das Tool *ProvViz* vorgestellt. Es wurde als Webanwendung mit React implementiert und stellt einen Editor für PROV dar. Zur Bearbeitung stehen ein Texteditor und ein graphischer Editor zur Verfügung, dabei werden fünf Notationen unterstützt. In [WM21] schreiben Werner und Moreau, dass bisherige Tools zur Provenance-Visualisierung nicht anfängerfreundlich wären, weshalb dieses Programm auf Nutzerfrendlichkeit ausgelegt sei. Außerdem habe ProvViz den Vorteil, dass zur Benutzung nur ein Browser benötigt wird und somit keine Softwareinstallation notwendig ist. Der Quellcode ist MIT-lizenziert und auf GitHub veröffentlicht.[6] Ausprobieren kann man das Tool auf der Webseite `https://provviz.com/`.

Die Abbildung 10.7 zeigt die Benutzeroberfläche des Programms, in welcher ein Teil des obigen Beispiels nachgebaut wurde. In der oberen blauen Leiste befindet sich auf der linken Seite der Name des Programms und zwei Menüfelder. Auf der rechten Seite kann das Dokumentenformat mit Hilfe eines Dropdown-Menüs gewählt werden. Es stehen unter anderem *PROV-XML* und *PROV-N* zur Verfügung, letzteres ist im Bild ausgewählt. Weiterhin kann man wählen, ob der Code, die Visualisierung oder beides angezeigt werden soll. Darunter befindet sich eine Tabliste mit den geöffneten Dokumenten. Den größten Teil des Bildes nimmt der Editor selbst ein. Auf der linken Seite wird der Quellcode in der gewählten Notation angezeigt. Die Visualisierung hingegen befindet sich auf der rechten Seite. Mehrere Schaltflächen stehen zur Verfügung, um den Graphen zu manipulieren oder diese herunterzuladen.

[5] `https://iitdbgroup.github.io/ProvenanceWeek2021/`
[6] `https://github.com/benwerner01/provviz-web`

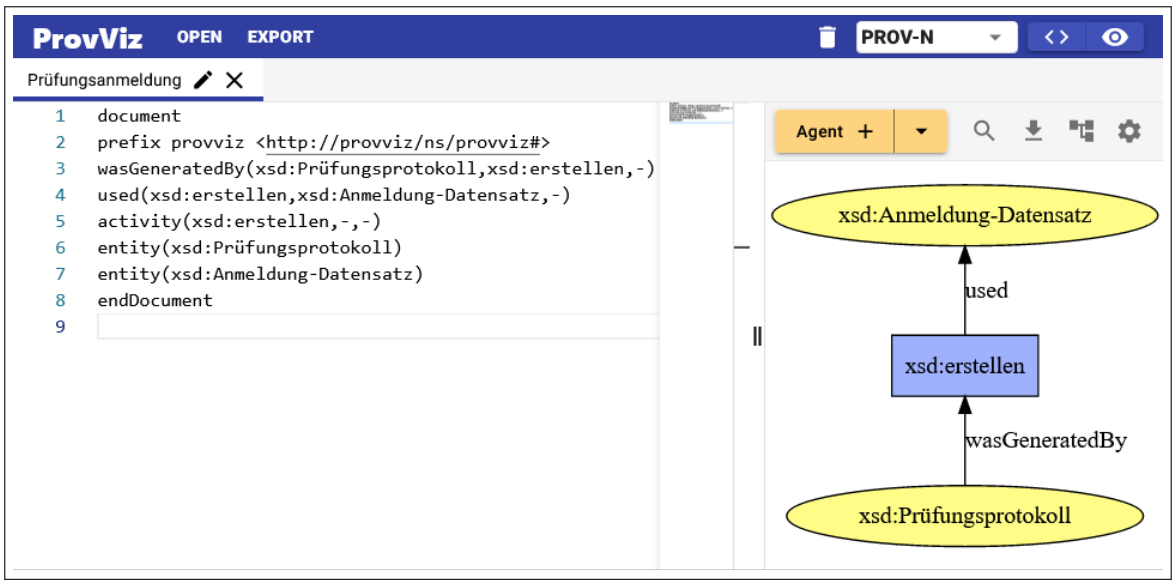

Abbildung 10.7: Benutzeroberfläche von ProvViz

ProvStore ProvStore [HM14a; HM14b] ist ein Webservice zum Teilen von PROV-Dokumenten. Es ist kein Programm wie die anderen Einträge in diesem Abschnitt. Dennoch soll es hier seine Erwähnung finden, weil es den PROV-Standard umsetzt und für die gemeinsame Arbeit mit diesem unterstützt. Laut der Veröffentlichung werden PROV-N, PROV-O, PROV-XML und PROV-JSON unterstützt. PROV-Dokumente können hochgeladen und dann mit ausgewählten Nutzern oder mit der Öffentlichkeit geteilt werden. Außerdem können externe Anwendungen auf Dokumente zugreifen. Zu diesem Zweck wurde eine REST-Schnittstelle[7] eingebaut. Weiterhin werden unterschiedliche Arten der Visualisierung unterstützt. Der Service ist unter der Adresse `https://openprovenance.org/store/` erreichbar.

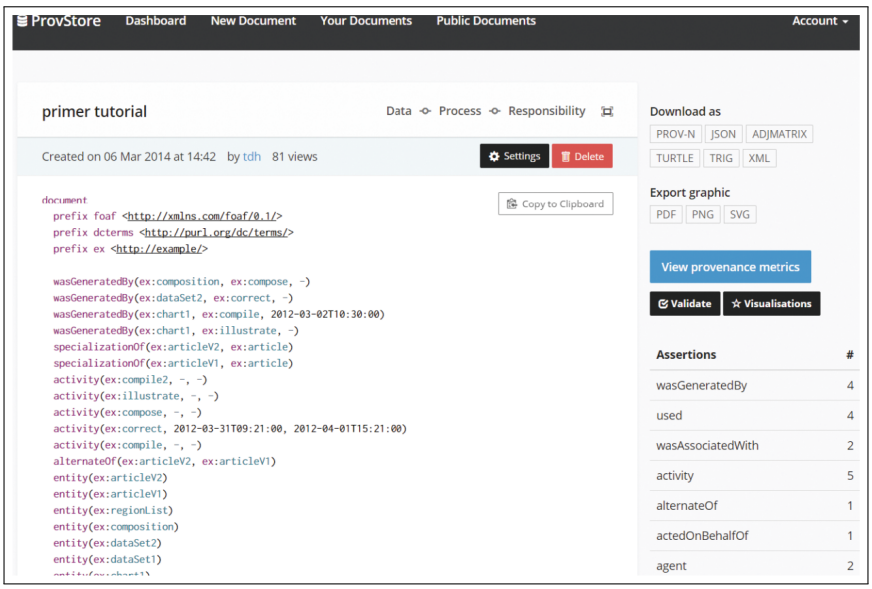

Abbildung 10.8: Bildschirmfoto von ProvStore mit einem PROV-Dokument [HM14a]

[7] `https://openprovenance.org/store/help/api/`

11. Zusammenfassung

Verschiedene Aspekte des Themengebietes Provenance wurden in dieser Literaturarbeit beleuchtet. In diesem Bericht wird eine Hierarchie von Provenance-Arten vorgestellt, die Data-Provenance, Workflow-Provenance, Information-System-Provenance und die allgemeinste Provenance-Art, d. h. Metadata-Provenance, umfasst. Dieser Bericht beschreibt den Status der Provenance-Arten und geht auf die Fragen ein, warum Provenance nützlich ist und wie Provenance-Daten für verschiedene Arten von Prozessen aufgezeichnet werden können. Jeder Provenance-Art, vom allgemeinsten Art (Metadaten-Provenance) bis zur spezifischsten Art (Daten-Provenance), wurde anhand von Studentenbeispielen untersucht. Außerdem wurde die Anwendung der Provenance auch im Hinblick auf drei Hauptkonzepte untersucht: Verständlichkeit, Reproduzierbarkeit und Qualität. Die Rolle der Provenance in verschiedenen Bereichen wie Verständlichkeit, die Leistung bei der Verdeutlichung der Ergebnisse und Prozesse für das Publikum, die Reproduzierbarkeit der Ergebnisse, die Rückverfolgbarkeit der Ergebnisse durch Aufzeichnung der verschiedenen Prozessstufen und die Überprüfung der Qualität der Ergebnisse sowie Prozesse wurden ausführlich beschrieben. Die verschiedenen Aufgaben, die mithilfe der Workflow-Provenance durchgeführt werden, umfassen drei verschiedene Dimensionen, die sich durch ihren Umfang, ihre Granularität und ihre Form unterscheiden. Vier Anwendungsbereiche der Workflow-Provenance wurden beschrieben: Wissenschaft, Wirtschaft, Datenanalyse und allgemeine Programmierung. Die Arten von Granularität sowie die Formen, die jede der genannten Domänen betreffen, wurden diskutiert. Als Nächstes wurde die Data Provenance vorgestellt. Auf die Provenance-Halbringe wurde dabei besonders eingegangen, da diese die Grundlage für die in der ***how***-Provenance verwendeten Provenance-Polynome bilden. Außerdem wurde auf die Provenance für Aggregationen, welche durch eine Erweiterung der Provenance-Halbringe dargestellt wird, eingegangen. Aber auch die Fragen ***why***, ***where*** und ***why-not*** wurden vorgestellt und jeweils an einem Beispiel erklärt. Im Abschnitt Provenance-Tools wurden bisherige Untersuchungen zu Tools wiedergegeben und ein Überblick in das Tool *Kepler* verschafft. Anhand eines Beispiels wurde dabei gezeigt, wie in Kepler sogenannte *Scientific Workflows* angelegt werden können, welche Probleme dabei auftreten können und wie sie sich lösen lassen. Danach gab es eine Einführung in PROV, einem W3C-Standard. Konzept sowie Notationen wurden am Beispiel einer Prüfungsanmeldung erklärt. Auch Tools, welche diesen Standard umsetzen, kommen hier nicht zu kurz. Drei Programme und ein Online-Repository stellte dieses Kapitel vor.

12. Quellenverzeichnis

[WS97] Allison Woodruff und Michael Stonebraker. „Supporting Fine-grained Data Lineage in a Database Visualization Environment“. In: *ICDE*. IEEE Computer Society, 1997, S. 91–102.

[CWW00] Yingwei Cui, Jennifer Widom und Janet L. Wiener. „Tracing the lineage of view data in a warehousing environment“. In: *ACM Trans. Database Syst.* 25.2 (2000), S. 179–227. DOI: 10.1145/357775.357777. URL: https://doi.org/10.1145/357775.357777.

[BKT01] Peter Buneman, Sanjeev Khanna und Wang Chiew Tan. „Why and Where: A Characterization of Data Provenance“. In: *Database Theory - ICDT 2001, 8th International Conference, London, UK, January 4-6, 2001, Proceedings*. Hrsg. von Jan Van den Bussche und Victor Vianu. Bd. 1973. Lecture Notes in Computer Science. Springer, 2001, S. 316–330. DOI: 10.1007/3-540-44503-X_20. URL: https://doi.org/10.1007/3-540-44503-X_20.

[Mun+06] Kiran-Kumar Muniswamy-Reddy, David A. Holland, Uri Braun und Margo I. Seltzer. „Provenance-Aware Storage Systems“. In: *Proceedings of the 2006 USENIX Annual Technical Conference, Boston, MA, USA, May 30 - June 3, 2006*. Hrsg. von Atul Adya und Erich M. Nahum. USENIX, 2006, S. 43–56. URL: http://www.usenix.org/events/usenix06/tech/muniswamy-reddy.html.

[The06] Olivier Thereaux. *Choose URIs wisely*. 24. Nov. 2006. URL: https://www.w3.org/QA/Tips/uri-choose (besucht am 26. 08. 2021).

[GKT07] Todd J. Green, Gregory Karvounarakis und Val Tannen. „Provenance semirings“. In: *Proceedings of the Twenty-Sixth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 11-13, 2007, Beijing, China*. Hrsg. von Leonid Libkin. ACM, 2007, S. 31–40. DOI: 10.1145/1265530.1265535. URL: https://doi.org/10.1145/1265530.1265535.

[DF08] Susan B. Davidson und Juliana Freire. „Provenance and scientific workflows: challenges and opportunities“. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*. Hrsg. von Jason Tsong-Li Wang. ACM, 2008, S. 1345–1350. DOI: 10.1145/1376616.1376772. URL: https://doi.org/10.1145/1376616.1376772.

[CCT09] James Cheney, Laura Chiticariu und Wang Chiew Tan. „Provenance in Databases: Why, How, and Where“. In: *Found. Trends Databases* 1.4 (2009), S. 379–474. DOI: 10.1561/1900000006. URL: https://doi.org/10.1561/1900000006.

[ADT11] Yael Amsterdamer, Daniel Deutch und Val Tannen. „Provenance for Aggregate Queries“. In: *CoRR* abs/1101.1110 (2011). arXiv: 1101.1110. URL: http://arxiv.org/abs/1101.1110.

[Mor11] Luc Moreau. „Provenance-based reproducibility in the Semantic Web“. In: *J. Web Semant.* 9.2 (2011), S. 202–221. DOI: 10.1016/j.websem.2011.03.001. URL: https://doi.org/10.1016/j.websem.2011.03.001.

[Gro+12] Paul Groth, Yolanda Gil, James Cheney und Simon Miles. „Requirements for Provenance on the Web“. In: *Int. J. Digit. Curation* 7.1 (2012), S. 39–56. DOI: 10.2218/ijdc.v7i1.213. URL: https://doi.org/10.2218/ijdc.v7i1.213.

[GM13a] Yolanda Gil und Simon Miles. *PROV Model Primer*. 30. Apr. 2013. URL: https://www.w3.org/TR/2013/NOTE-prov-primer-20130430/ (besucht am 14. 07. 2021).

[GM13b] Paul Groth und Luc Moreau. *PROV-Overview*. 30. Apr. 2013. URL: https://www.w3.org/TR/2013/NOTE-prov-overview-20130430/ (besucht am 14. 07. 2021).

[LSM13] Timothy Lebo, Satya Sahoo und Deborah McGuinness. *PROV-O: The PROV Ontology*. 30. Apr. 2013. URL: https://www.w3.org/TR/2013/REC-prov-o-20130430/ (besucht am 14. 09. 2021).

[MBC13] Paolo Missier, Khalid Belhajjame und James Cheney. „The W3C PROV Family of Specifications for Modelling Provenance Metadata“. In: EDBT '13. Genoa, Italy: Association for Computing Machinery, 2013, S. 773–776. ISBN: 9781450315975. DOI: 10.1145/2452376.2452478. URL: https://doi.org/10.1145/2452376.2452478.

[MM13a] Luc Moreau und Paolo Missier. *PROV-DM: The PROV Data Model*. 30. Apr. 2013. URL: https://www.w3.org/TR/2013/REC-prov-dm-20130430/ (besucht am 14. 07. 2021).

[MM13b] Luc Moreau und Paolo Missier. *PROV-N: The Provenance Notation*. 30. Apr. 2013. URL: https://www.w3.org/TR/2013/REC-prov-n-20130430/ (besucht am 30. 09. 2021).

[Nie+13] Tom De Nies, Sara Magliacane, Ruben Verborgh, Sam Coppens, Paul Groth, Erik Mannens und Rik Van de Walle. „Git2PROV: Exposing Version Control System Content as W3C PROV“. In: *Proceedings of the ISWC 2013 Posters & Demonstrations Track, Sydney, Australia, October 23, 2013*. Hrsg. von Eva Blomqvist und Tudor Groza. Bd. 1035. CEUR Workshop Proceedings. CEUR-WS.org, 2013, S. 125–128. URL: http://ceur-ws.org/Vol-1035/iswc2013_demo_32.pdf.

[Obj13] Object Management Group. *Business Process Model and Notation (BPMN). Version 2.0.2*. Dez. 2013. URL: https://www.omg.org/spec/BPMN/2.0.2/PDF (besucht am 30. 09. 2021).

[Sue+13] Chun Hui Suen, Ryan K. L. Ko, Yu Shyang Tan, Peter Jagadpramana und Bu-Sung Lee. „S2Logger: End-to-End Data Tracking Mechanism for Cloud Data Provenance“. In: *12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2013 / 11th IEEE International Symposium on Parallel and Distributed Processing with Applications, ISPA-13 / 12th IEEE International Conference on Ubiquitous Computing and Communications, IUCC-2013, Melbourne, Australia, July 16-18, 2013*. IEEE Computer Society,

2013, S. 594–602. DOI: 10.1109/TrustCom.2013.73. URL: https://doi.org/10.1109/TrustCom.2013.73.

[HG14] Rinke Hoekstra und Paul Groth. „PROV-O-Viz - Understanding the Role of Activities in Provenance“. In: *Provenance and Annotation of Data and Processes - 5th International Provenance and Annotation Workshop, IPAW 2014, Cologne, Germany, June 9-13, 2014. Revised Selected Papers*. Hrsg. von Bertram Ludäscher und Beth Plale. Bd. 8628. Lecture Notes in Computer Science. Springer, 2014, S. 215–220. DOI: 10.1007/978-3-319-16462-5_18. URL: https://doi.org/10.1007/978-3-319-16462-5_18.

[HM14a] Trung Dong Huynh und Luc Moreau. „ProvStore: A Public Provenance Repository“. In: *Provenance and Annotation of Data and Processes - 5th International Provenance and Annotation Workshop, IPAW 2014, Cologne, Germany, June 9-13, 2014. Revised Selected Papers*. Hrsg. von Bertram Ludäscher und Beth Plale. Bd. 8628. Lecture Notes in Computer Science. Springer, 2014, S. 275–277. DOI: 10.1007/978-3-319-16462-5_32. URL: https://doi.org/10.1007/978-3-319-16462-5_32.

[HM14b] Trung Dong Huynh und Luc Moreau. „ProvStore: a public provenance repository“. In: *5th International Provenance and Annotation Workshop (IPAW'14)*. Juni 2014. URL: https://eprints.soton.ac.uk/365509/.

[Obj14] Object Management Group. *About the Business Process Model and Notation Specification Version 2.0.2*. Jan. 2014. URL: https://www.omg.org/spec/BPMN/2.0.2/About-BPMN/ (besucht am 30. 09. 2021).

[Cow15] Charles Cowart. *Getting Started Guide. Version 2.5*. Okt. 2015. URL: https://code.kepler-project.org/code/kepler-docs/trunk/outreach/documentation/shipping/2.5/getting-started-guide.pdf (besucht am 30. 09. 2021).

[Her15] Melanie Herschel. „A Hybrid Approach to Answering Why-Not Questions on Relational Query Results“. In: *ACM J. Data Inf. Qual.* 5.3 (2015), 10:1–10:29. DOI: 10.1145/2665070. URL: https://doi.org/10.1145/2665070.

[LP15] „Provenance and Annotation of Data and Processes - 5th International Provenance and Annotation Workshop, IPAW 2014, Cologne, Germany, June 9-13, 2014. Revised Selected Papers“. In: Lecture Notes in Computer Science 8628 (2015). Hrsg. von Bertram Ludäscher und Beth Plale. DOI: 10.1007/978-3-319-16462-5. URL: https://doi.org/10.1007/978-3-319-16462-5.

[Nie+15] Tom De Nies, Io Taxidou, Anastasia Dimou, Ruben Verborgh, Peter M. Fischer, Erik Mannens und Rik Van de Walle. „Towards Multi-level Provenance Reconstruction of Information Diffusion on Social Media“. In: *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*. Hrsg. von James Bailey, Alistair Moffat, Charu C. Aggarwal, Maarten de Rijke, Ravi Kumar, Vanessa Murdock, Timos K. Sellis und Jeffrey Xu Yu. ACM, 2015, S. 1823–1826. DOI: 10.1145/2806416.2806642. URL: https://doi.org/10.1145/2806416.2806642.

[HH16] Melanie Herschel und Marcel Hlawatsch. „Provenance: On and Behind the Screens“. In: *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*. Hrsg. von Fatma Özcan, Georgia Koutrika und Sam Madden. ACM, 2016, S. 2213–2217. DOI: 10.1145/2882903.2912568. URL: https://doi.org/10.1145/2882903.2912568.

[Heu16] Andreas Heuer. *Ringvorlesung: Forschung @ DBIS*. Vorlesungsfolien, Wintersemester 2015/16. Universität Rostock, 2016.

[Rag+16] Eric D. Ragan, Alex Endert, Jibonananda Sanyal und Jian Chen. „Characterizing Provenance in Visualization and Data Analysis: An Organizational Framework of Provenance Types and Purposes“. In: *IEEE Trans. Vis. Comput. Graph.* 22.1 (2016), S. 31–40. DOI: 10.1109/TVCG.2015.2467551. URL: https://doi.org/10.1109/TVCG.2015.2467551.

[Wil+16] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne u. a. „The FAIR Guiding Principles for scientific data management and stewardship“. In: *Scientific data* 3 (2016).

[Aug17] Tanja Auge. „Umsetzung von Provenance-Anfragen in Big-Data-Analytics-Umgebungen“. Masterarbeit. Universität Rostock, Sep. 2017. URL: http://eprints.dbis.informatik.uni-rostock.de/880/1/MasterarbeitInformatik.pdf.

[GT17] Todd J. Green und Val Tannen. „The Semiring Framework for Database Provenance“. In: *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2017, Chicago, IL, USA, May 14-19, 2017*. Hrsg. von Emanuel Sallinger, Jan Van den Bussche und Floris Geerts. ACM, 2017, S. 93–99. DOI: 10.1145/3034786.3056125. URL: https://doi.org/10.1145/3034786.3056125.

[HDB17] Melanie Herschel, Ralf Diestelkämper und Houssem Ben Lahmar. „A survey on provenance: What for? What form? What from?“ In: *VLDB J.* 26.6 (2017), S. 881–906. DOI: 10.1007/s00778-017-0486-1. URL: https://doi.org/10.1007/s00778-017-0486-1.

[BHK18] Felix Bartusch, Maximilian Hanussek und Jens Krüger. „Automatic Generation of Provenance Metadata during Execution of Scientific Workflows“. In: CEUR Workshop Proceedings 2357 (2018). Hrsg. von Malcolm P. Atkinson und Sandra Gesing. URL: http://ceur-ws.org/Vol-2357/paper8.pdf.

[FC18] Juliana Freire und Fernando Seabra Chirigati. „Provenance and the Different Flavors of Reproducibility“. In: *IEEE Data Eng. Bull.* 41.1 (2018), S. 15–26. URL: http://sites.computer.org/debull/A18mar/p15.pdf.

[Hen18] Shawn Lawton Henry. *Referencing and Linking to WAI Guidelines and Technical Documents*. 10. Dez. 2018. URL: https://www.w3.org/WAI/standards-guidelines/linking/ (besucht am 26. 08. 2021).

[KMF18] David Koop, Marta Mattoso und Juliana Freire. „Provenance in Workflows“. In: *Encyclopedia of Database Systems, Second Edition*. Hrsg. von Ling Liu und M. Tamer Özsu. Springer, 2018. DOI: 10.1007/978-1-4614-8265-9_80745. URL: https://doi.org/10.1007/978-1-4614-8265-9_80745.

[ACM20] ACM. *Artifact Review and Badging*. 24. Aug. 2020. URL: https://www.acm.org/publications/policies/artifact-review-badging (besucht am 30. 09. 2021).

[Fla+21] Rocco Flach, Maximilian Lamster, Chris Röhrs, Nic Scharlau und Tanja Auge. *Provenance Tools*. Semesterarbeit. Apr. 2021.

[Gle+21] Lars Gleim, Jan Pennekamp, Liam Tirpitz, Sascha Welten, Florian Brillowski und Decker Stefan. „FactStack“. In: *BTW 2021*. Hrsg. von Kai-Uwe Sattler, Melanie Herschel und Wolfgang Lehner. Gesellschaft für Informatik, Bonn, 2021, S. 371–395. DOI: 10.18420/btw2021-20.

[WM21] Ben Werner und Luc Moreau. „ProvViz: An Intuitive Prov Editor and Visualiser“. In: *Provenance and Annotation of Data and Processes - 8th and 9th International Provenance and Annotation Workshop, IPAW 2020 + IPAW 2021, Virtual Event, July 19-22, 2021, Proceedings*. Hrsg. von Boris Glavic, Vanessa Braganholo und David Koop. Bd. 12839. Lecture Notes in Computer Science. Springer, 2021, S. 231–236. DOI: 10.1007/978-3-030-80960-7_18. URL: https://doi.org/10.1007/978-3-030-80960-7_18.

[CAS] CASRAI. *Provenance metadata*. URL: https://casrai.org/term/provenance-metadata/ (besucht am 12. 07. 2021).

[Dat] The Data Visualisation Catalogue. *Network Diagram*. URL: https://datavizcatalogue.com/methods/network_diagram.html (besucht am 30. 09. 2021).

[Kep] The Kepler Project Authors. *The Kepler Project*. URL: https://kepler-project.org/index.html (besucht am 30. 09. 2021).

13. Abbildungsverzeichnis

14. Tabellenverzeichnis

15. Liste der Befehle

A. Studierenden-Beispiel

A.1. Relationen der Beispieldatenbank

s_id	student_id	lastname	firstname	study_course
S_1	1	Moore	Donald	Computer Science for Teaching
S_2	2	Morgan	Sarah	Mathematics
S_3	3	Wood	Jack	Electrical Engineering
S_4	4	Harrison	Elisabeth	Computer Science
S_5	5	Williams	John	Computer Science
S_6	6	William	Mary	Computer Science
S_7	7	Smith	Jack	Electrical Engineering
S_8	8	John	Jennifer	Theoretical Computer Science

Tabelle A.1: Table STUDENTS

c_id	course_nr	title
C_1	001	Database Systems and Data Science
C_2	002	Operating Systems
C_3	003	Artificial Intelligence
C_4	004	Probability Theory
C_5	005	Algorithms and Data Structures
C_6	006	Object-Oriented Programming
C_7	007	Law and Computer Science
C_8	008	Data Warehouses and Business Intelligence
C_9	009	Networks and Cybersecurity

Tabelle A.2: Table COURSES

l_id	course_nr	fullname
$L_{1.1}$	001	Professor A
$L_{1.2}$	001	Lecturer A
L_2	002	Professor B
L_3	003	Professor C
L_4	004	Professor D
L_5	005	Lecturer B
L_6	006	Professor D
L_7	007	Professor A
L_8	008	Professor E
L_9	009	Professor A

Tabelle A.3: Table LECTURERS

p_id	course_nr	student_id
P_1	001	1
P_2	001	2
P_3	001	4
P_4	001	5
P_5	002	1
P_6	002	2
P_7	002	3
P_8	002	4
P_9	002	5
P_{10}	002	6
P_{11}	002	7
P_{12}	003	1
P_{13}	004	3
P_{14}	004	5
P_{15}	004	7
P_{16}	005	4
P_{17}	005	7
P_{18}	006	4
P_{19}	006	5
P_{20}	007	1
P_{21}	007	3
P_{22}	007	5
P_{23}	008	3
P_{24}	009	1
P_{25}	009	4
P_{26}	009	5

Tabelle A.4: Table PARTICIPANTS

g_id	course_nr	student_id	semester	grade
G_1	001	1	SS 16	2.0
G_2	001	2	SS 16	1.7
G_3	001	4	SS 16	1.7
G_4	001	5	SS 16	3.0
G_5	002	1	WS 15/16	3.7
G_6	002	2	WS 14/15	1.3
G_7	002	3	WS 14/15	2.3
G_8	002	4	WS 15/16	1.0
G_9	002	5	WS 15/16	1.3
G_{10}	002	6	WS 15/16	2.0
G_{11}	002	7	WS 15/16	3.3
G_{12}	003	1	WS 16/17	1.0
G_{13}	004	3	WS 16/17	1.3
G_{14}	004	5	WS 16/17	3.0
G_{15}	005	4	SS 17	2.7
G_{16}	005	7	SS 17	1.7
G_{17}	006	4	SS 17	2.7
G_{18}	006	5	SS 17	4.0
G_{19}	007	1	SS 16	2.3
G_{20}	007	3	SS 16	1.7
G_{21}	009	1	SS 16	3.3
G_{22}	009	5	SS 15	5.0
G_{23}	009	5	SS 16	2.7

Tabelle A.5: Table GRADES

A.2. Studierendenbeispiel in PROV-N

```
prefix ex <https://example.org/>

entity(ex:Student-Datensatz)
entity(ex:Formular)
entity(ex:Bestätigtes-Formular)
entity(ex:Zugangsvoraussetzungen)
entity(ex:Anmeldung-Datensatz)
entity(ex:Prüfungsprotokoll)

activity(ex:prüfen1)
activity(ex:prüfen2)
activity(ex:erstellen, 2021-07-15T12:34:56, 2021-07-15T12:34:59)

agent(ex:Herr-Müller, [ prov:type='prov:Person' ])
agent(ex:Prüfungsamt, [ prov:type='prov:Organization' ])
agent(ex:Prüfungssoftware, [ prov:type='prov:SoftwareAgent' ])

used(ex:prüfen1, ex:Student-Datensatz, -)
used(ex:prüfen1, ex:Formular, -)
used(ex:prüfen2, ex:Zugangsvoraussetzungen, -)
used(ex:prüfen2, ex:Bestätigtes-Formular, -)
used(ex:erstellen, ex:Anmeldung-Datensatz, -)

wasGeneratedBy(ex:Bestätigtes-Formular, ex:prüfen1, -)
wasGeneratedBy(ex:Anmeldung-Datensatz, ex:prüfen2, -)
wasGeneratedBy(ex:Prüfungsprotokoll, ex:erstellen, -)

wasAssociatedWith(ex:prüfen1, ex:Herr-Müller, -)
wasAssociatedWith(ex:prüfen2, ex:Prüfungssoftware, -)
wasAssociatedWith(ex:erstellen, ex:Prüfungssoftware, -)

wasAttributedTo(ex:Prüfungsprotokoll, ex:Prüfungssoftware)
wasAttributedTo(ex:Anmeldung-Datensatz, ex:Prüfungssoftware)

actedOnBehalfOf(ex:Herr-Müller, ex:Prüfungsamt, -)
```