

Bachelorarbeit

Pascal Heinsohn

Vergleichende Evaluation von Open-Source Software für
Forschungsdatenmanagement

Pascal Heinsohn

Vergleichende Evaluation von Open-Source Software für Forschungsdatenmanagement

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Angewandte Informatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Olaf Zukunft
Zweitgutachter: Prof. Dr. Martin Becke

Eingereicht am: 15. Juli 2020

Pascal Heinsohn

Thema der Arbeit

Vergleichende Evaluation von Open-Source Software für Forschungsdatenmanagement

Stichworte

Datenmanagement, Open Source, Evaluation

Kurzzusammenfassung

Diese Arbeit befasst sich mit der Evaluierung zweier Open-Source-Systeme für Forschungsdatenmanagement. Hierfür werden Anforderungen für zwei Anwendungsfälle erhoben, Defizite bezüglich der Anforderungen identifiziert, eine Erweiterung für eines der Systeme konzipiert und implementiert und schließlich beide Systeme bewertet.

Pascal Heinsohn

Title of Thesis

A comparative evaluation of Open-Source software for research data management

Keywords

Data Management, Open Source, Evaluation

Abstract

This thesis deals with the evaluation of two Open Source systems for research data management. For that purpose requirements for two use cases are collected, deficits regarding those requirements are identified, an extension for one of the two systems is designed and implemented and finally both systems are evaluated.

Inhaltsverzeichnis

Abbildungsverzeichnis	vi
Tabellenverzeichnis	vii
1 Einleitung	1
2 Grundlagen	2
2.1 Research Data Management	2
2.2 CKAN	3
2.2.1 Was ist mit CKAN möglich?	3
2.2.2 Struktur der Plattform	4
2.3 Dataverse	5
2.3.1 Was ist mit Dataverse möglich?	6
2.3.2 Struktur der Plattform	7
2.4 Evaluationsmethodik	9
3 Analyse	14
3.1 Anforderungen an die Systeme	14
3.1.1 Situation im HAW Big Data Labor	14
3.1.2 Situation der Hamburg City Health Study	16
3.1.3 Voraussetzungen für die Bewertung der Systeme	17
3.2 Identifikation von Defiziten bzgl. der Anforderungen	19
3.3 Beheben der Defizite durch Implementation von Extensions	20
3.3.1 Variablenkatalog für CKAN	20
3.3.2 Variablenkatalog für Dataverse	22
4 Entwurf einer Extension	24
4.1 Iteration 1: Wiederverwendung des DataExplorers	24
4.2 Iteration 2: Das Framework Tabulator	25

4.3	Iteration 3: Verwendung von “rohem” Recline.js mit CSV	26
5	Implementation einer Extension	28
5.1	Iteration 1: Wiederverwendung des DataExplorers	28
5.2	Iteration 2: Das Framework Tabulator	28
5.3	Iteration 3: Verwendung von “rohem” Recline.js mit CSV	29
6	Bewertung	32
6.1	Definition eines Qualitätsmodells	32
6.2	Durchführung der Evaluation	37
6.2.1	CKAN	37
6.2.1.1	Functional Suitability (Funktionelle Eignung)	38
6.2.1.2	Performance Efficiency (Performanz)	43
6.2.1.3	Compatibility (Kompatibilität)	44
6.2.1.4	Usability (Benutzbarkeit)	45
6.2.1.5	Reliability (Zuverlässigkeit)	50
6.2.1.6	Security (Sicherheit)	51
6.2.1.7	Maintainability (Wartbarkeit)	54
6.2.1.8	Portability (Portierbarkeit)	56
6.2.2	Dataverse	58
6.2.2.1	Functional Suitability (Funktionelle Eignung)	58
6.2.2.2	Performance Efficiency (Performanz)	63
6.2.2.3	Compatibility (Kompatibilität)	63
6.2.2.4	Usability (Benutzbarkeit)	64
6.2.2.5	Reliability (Zuverlässigkeit)	67
6.2.2.6	Security (Sicherheit)	68
6.2.2.7	Maintainability (Wartbarkeit)	71
6.2.2.8	Portability (Portierbarkeit)	72
6.3	Vergleich der Evaluationsergebnisse und Abschluss	73
7	Zusammenfassung und Ausblick	75
	Literaturverzeichnis	76
	Selbstständigkeitserklärung	80

Abbildungsverzeichnis

2.1	Struktur der Entitäten in CKAN	4
2.2	Dataverse Usability: Erfolgsverteilung bei Szenario 1 [31]	6
2.3	Dataverse Usability: Erfolgsverteilung bei Szenario 2 [31]	7
2.4	Dataverse: hierarchische Struktur [29]	8
2.5	ISO/IEC 25040: Fünf Schritte einer Software-Evaluation [15]	9
2.6	ISO/IEC 25010: Gruppierungen von Kriterien [15]	11
3.1	CKAN: simple und übersichtliche Darstellung von Tabellendaten	20
3.2	CKAN: Mögliche Platzierung des Variablenkatalog-Buttons	22
3.3	Dataverse: Verlinkung des Variablenkatalogs im Root-Dataverse	23
4.1	CKAN-Extension: MVC-ähnliches Schema	25
4.2	CKAN-Extension: Entwurf mit Tabulator	26
4.3	CKAN-Extension: Entwurf mit Recline.js	27
5.1	Recline.js: Darstellung & Bearbeitung einer Tabelle im Browser	30
5.2	CKAN: erfolgreich implementierte Extension	31
6.1	CKAN: Nicht sehr intuitive Lokation der Benutzerübersicht	41
6.2	CKAN: Responsive Design ermöglicht Verwendung an Mobilgeräten	48
6.3	CKAN: Buttons in der Menüleiste bestehen nur aus Symbolen	49
6.4	CKAN: angenehme Farben sorgen für Wohlbefinden bei der Verwendung	49
6.5	CKAN: Überblick über die Architektur [22]	55
6.6	Dataverse: die acht Standard-Benutzerrollen	58
6.7	Dataverse: Responsive Design ermöglicht Verwendung an Mobilgeräten	66
6.8	Dataverse: gekürzte Log-Tabelle “actionlogrecord”	69

Tabellenverzeichnis

2.1	ISO/IEC 25010: Qualitätsmerkmale und deren Definitionen [15]	13
3.1	Für die Kategorie Functional Suitability benötigte Ziele und Aufgaben . .	17
6.1	Die vier Schritte dieser Evaluation	32
6.2	Punktesystem für die Bewertung	33
6.3	Definition von Metriken	34
6.4	CKAN: Functional Completeness	43
6.5	CKAN: Resource Utilization	44
6.6	CKAN: Interoperability	45
6.7	CKAN: Operability	47
6.8	CKAN: Integrity	52
6.9	CKAN: Non-repudiation	53
6.10	CKAN: Installability	57
6.11	Dataverse: Functional Completeness	62
6.12	Dataverse: Resource Utilization	63
6.13	Dataverse: Operability	65
6.14	Dataverse: Integrity	69
6.15	Dataverse: Non-repudiation	70
6.16	Vergleich der Evaluationsergebnisse	73

1 Einleitung

In dieser Arbeit werden die beiden Open-Source-Softwaresysteme CKAN und Dataverse bewertet und miteinander verglichen. Die beiden Projekte sind aus der Open-Science-Bewegung hervorgegangen und sollen Forschenden dabei helfen, ihre erhobenen Daten an einem zentralen Ort geordnet zu sammeln und mit anderen Interessierten zu teilen.

Die Motivation für dieses Thema rührt daher, dass sowohl das Big-Data-Labor der Hochschule für Angewandte Wissenschaften (HAW) Hamburg, an der diese Arbeit entsteht, als auch der Arbeitgeber des Autors, das Epidemiologische Studienzentrum des Universitätsklinikums Hamburg-Eppendorf (UKE) ein Interesse an der Herstellung einer geordneten Sammlung erhobener Forschungsdaten haben.

Das Ziel der Arbeit ist es, die beiden Systeme zu testen und zu bewerten sowie für beide Anwendungsfälle den besser geeigneten der beiden Kandidaten zu bestimmen. Hierfür werden beide Systeme auf gleichen virtuellen Maschinen installiert, nach demselben Schema in einem breit aufgestellten Spektrum von Kriterien, die dem Industriestandard entsprechen, bewertet und schließlich wird eine Empfehlung für beide Anwendungsfälle ausgesprochen.

Kapitel 2 führt in die Grundlagen des Themas ein und die beiden Systeme werden grob vorgestellt. Zudem wird hier die in leichter Abwandlung schließlich verwendete Evaluationsmethodik erläutert. In Kapitel 3 werden Anforderungen an die Systeme definiert, die Ausgangssituationen der beiden Anwendungsfälle sondiert sowie Voraussetzungen für die Bewertung beider Systeme festgelegt. In den Kapiteln 4 und 5 wird auf iterative Weise eine Erweiterung für eines der beiden Systeme konstruiert und implementiert, während in Kapitel 6 die tatsächliche Bewertung der beiden Systeme erfolgt. Kapitel 7 fasst die Arbeit abschließend noch einmal zusammen und stellt eine potenzielle Fortführung in Aussicht.

2 Grundlagen

In diesem Kapitel wird erläutert, was die Schwierigkeit beim Forschungsdatenmanagement ist. Zudem werden die beiden zu untersuchenden Systeme sowie der Industriestandard für die Evaluierung von Softwaresystemen vorgestellt.

2.1 Research Data Management

Der Begriff “Datenmanagement” ist ein Sammelbegriff für alle Tätigkeiten und Aufgaben, die Daten als Ressource verwenden. Diese Tätigkeiten gehen vom Entwickeln von Datenstrukturen über das Erheben der Daten und Qualitätskontrolle bis hin zur Veröffentlichung der Daten. [9] Als “Research Data Management” kann man also die Menge aller Tätigkeiten und Aufgaben, die mit Forschungsdaten zu tun haben, bezeichnen. Forschungsdaten erfordern zum Teil einen besonderen Umgang, da z.B. persönliche medizinische Daten strengen Datenschutzauflagen unterliegen.

Das JISC (Joint Information Systems Committee), eine “gemeinnützige Organisation zur Förderung digitaler Technologien in Forschung und Lehre” [17] aus Großbritannien, beschrieb im Jahre 2011 das Research Data Management als eine der dringlichsten Herausforderungen des Hochschul- und Forschungssektors. Aus öffentlich finanzierten Projekten hervorgehende Forschungsdaten seien ein öffentliches Gut und sollten zur Validierung und Wiederverwendung zugänglich sein. Hochschuleinrichtungen stünden aus verschiedenen Gründen unter Druck, Dienste und Infrastrukturen für Forschungsdatenmanagement bereitzustellen: [21]

- die wachsenden Möglichkeiten, gemeinsam mit anderen Institutionen datenintensivere und offenere Forschung durchzuführen,
- die wachsenden Anforderungen von Förderern und Partnern,
- die zunehmende Sorge um Forschungstransparenz und -integrität,

- die Angst vor Datenverlust. [21]

2.2 CKAN

Die CKAN Association beschreibt ihr Produkt, dessen Name für “Comprehensive Knowledge Archive Network” steht, auf der eigenen Website selbst als den Weltmarktführer auf dem Gebiet der Open Source Datenportal-Plattformen. Das Produkt sei an Regierungen, Firmen und Organisationen adressiert, welche Daten kostenlos verfügbar machen wollen. CKAN wird laut Website von verschiedenen Regierungen (Kanada, USA, Australien) sowie z.B. dem European Data Portal genutzt. [5] Auf der “Features”-Seite wirbt die CKAN Association u.a. mit einer RPC-artigen API, einfacher Erweiterbarkeit und einem intuitiven Webinterface.

Die Entwicklung an der Software begann im März 2006, der erste Release folgte im Juli 2007. [35] Inzwischen gibt es weltweit mindestens 200 laufende Instanzen von CKAN. Dies ist die Anzahl der auf der Website registrierten Portale, aber die tatsächliche Zahl lässt sich natürlich dadurch, dass das Projekt Open Source auf Github angeboten wird, nicht nachvollziehen und liegt wahrscheinlich deutlich höher. [5]

2.2.1 Was ist mit CKAN möglich?

Die Betreiber zweier CKAN-Instanzen, einerseits die australische Regierung mit ihrer Plattform <https://data.gov.au> und andererseits eine niederländische Firma, Civity (<https://dataplatform.nl>), die es sich zum Ziel gemacht hat, Smart Cities voranzutreiben, stellen ihre Anwendungsfälle in zwei kurzen Fallstudien auf der CKAN-Website vor und erläutern, wieso sie sich für CKAN entschieden haben.

Das Open Data Portal der australischen Regierung, “the Australian Federal Open Data Portal” wurde 2013 ins Leben gerufen. Es dient dem Zweck, der Öffentlichkeit Datensätze von der Regierung zur freien Verfügung zu stellen. Da sich die initiale Lösung als WordPress-Instanz als unkomfortabel sowohl für Publizierende als auch für Endnutzer erwies, entschied man sich für eine spezialisiertere Lösung und ließ CKAN durch einen auf CKAN spezialisierten externen Dienstleister installieren und die Daten migrieren. [5] Dadurch, dass CKAN die Möglichkeit bietet, Datensätze automatisiert zu publizieren, werden einige Datensätze auf der Seite täglich aktualisiert. [5] Inzwischen befinden sich in dem Portal über 90.000 Datensätze. [7]

Die andere Fallstudie, von den Betreibern der dataplatform.nl, zeigt, dass auch eine andere Herangehensweise mit CKAN möglich ist. Die Betreiber stellen sowohl ein eigenes CKAN-Repository für einzelne niederländische Städte (z.B. utrecht.dataplatform.nl) als auch ein zentrales Repository bereit, in dem alle Daten zusammenlaufen. So hat jede Gemeinde ihre eigene Plattform, aber die Daten laufen auch gesamt niederländisch an einem Punkt zusammen. [5]

2.2.2 Struktur der Plattform

In einer CKAN-Instanz ohne Extensions oder anderweitige Individualisierungen gibt es vier verschiedene Arten von Entitäten: Organizations, Users, Datasets und Groups.

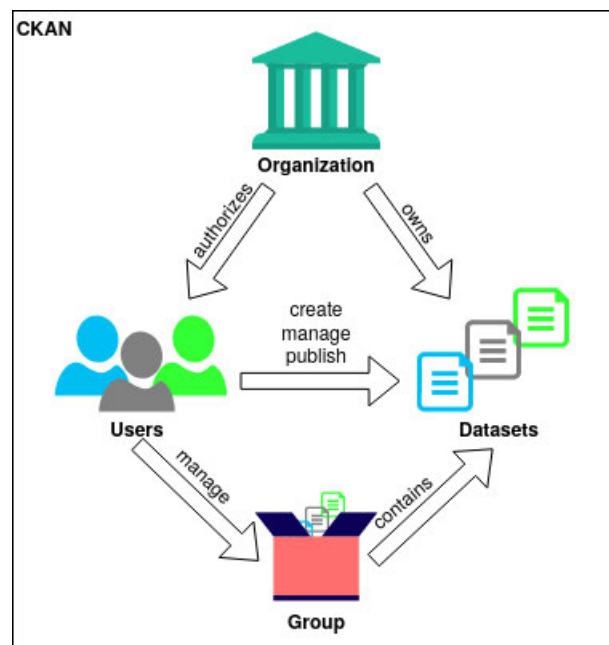


Abbildung 2.1: Struktur der Entitäten in CKAN

An der Spitze stehen die “Organizations”. In einer CKAN-Instanz kann es mehrere Organizations geben, wodurch sich z.B. verschiedene Projektgruppen darstellen lassen. Organizations verwalten mithilfe eines Berechtigungssystems ihre User und sind die Besitzer von Datasets. Je nach Rolle innerhalb einer Organization kann ein User verschiedene Aktionen ausführen:

- ein Member kann - im Vergleich zu einem User, der nicht Mitglied einer Organization ist - private Datasets einer Organization einsehen;
- ein Editor kann die bekannten CRUD¹-Operationen auf alle Datasets der Organization ausführen;
- und ein Admin kann zusätzlich noch die Mitglieder der Organization verwalten.

Ein Dataset hat immer einen Titel und kann zusätzlich verschiedene Metadaten (z.B. Beschreibung, Tags, Angaben zur Lizenz und der Autorenschaft) sowie mehrere zugehörige Dateien beinhalten. Diese Dateien können in allen verschiedenen Formaten und Größen (sofern nicht anders konfiguriert) hochgeladen werden. Einige dieser Formate (z.B. Bild- und Tabellenformate) lassen sich dann direkt in der Webapplikation in einer Vorschau anzeigen.

Die vierte Entitätsart ist die “Group”. In einer Group kann ein User mehrere Datasets sammeln, damit er zum Beispiel alle Datasets, die er für ein Projekt benutzt, gebündelt an einem Ort wiederfindet und nicht immer einzeln danach suchen muss. Der Name ist etwas missverständlich gewählt, “Collection” wäre beispielsweise passender und würde nicht mit einer Gruppe von Usern verwechselt werden.

2.3 Dataverse

Das Dataverse Project wurde vom Institute for Quantitative Social Science (IQSS) der Harvard University ins Leben gerufen und wird auf der Produktwebsite als Open Source Webapplikation zum Teilen, Aufbewahren, Zitieren, Erkunden und Analysieren von Forschungsdaten betitelt. Eine zentrale Eigenschaft von Dataverse sei es, die Aufgaben eines klassischen Archivars zu automatisieren. [8]

Das Projekt wurde ebenso wie CKAN im Jahre 2006 begonnen und 2007 erstmals veröffentlicht. [30] Aktuell gibt es laut der Dataverse-Website 56 laufende Instanzen weltweit, aber ebenso wie bei CKAN ist es hier nicht ganz nachvollziehbar, da auch dieses Projekt Open Source ist und sich sicher nicht jeder Betreiber einer Instanz bei Dataverse registriert hat. [8] Inzwischen wird das Projekt von freiwilligen Helfern auf der ganzen Welt mitentwickelt. Unter den auf der Website beworbenen prominenten Anwendern befinden sich hauptsächlich Universitäten und andere staatliche Forschungseinrichtungen,

¹create, read, update, delete

z.B. die Harvard University selbst, die Johns Hopkins Universität in Baltimore oder die Universität Heidelberg. [8]

2.3.1 Was ist mit Dataverse möglich?

Das Dataverse-Team hat die Plattform im Rahmen eines Usability-Tests durch ein unabhängiges Team bestehend aus Studenten und einem Professor des Masterstudiengangs “Library and Information Science” der Simmons University in Boston auf Gebrauchstauglichkeit testen lassen, um in zukünftigen Versionen eine bessere User Experience gewährleisten zu können. [31]

Im Rahmen ihrer Arbeit hat das Team zwei Szenarios entwickelt:

1. Das Benutzen der Plattform als jemand, der Daten sucht (“data user”). Die Probanden mussten in diesem Szenario Aufgaben lösen, bei denen es um grundsätzliche Funktionen von Dataverse ging: Das Benutzen der normalen und erweiterten Suche, Downloaden von bestimmten Dateien aus bestimmten Studien oder das Finden des “Fehler melden”-Buttons. [31]

Wie in Abbildung 2.2 zu sehen ist, konnten neun der insgesamt vierzehn Aufgaben ($\approx 64,3\%$) von mindestens der Hälfte der “Data User” mit Leichtigkeit bewältigt werden, während sogar alle Aufgaben eher (also trotz Schwierigkeiten) bewältigt als nicht bewältigt werden konnten. [31]

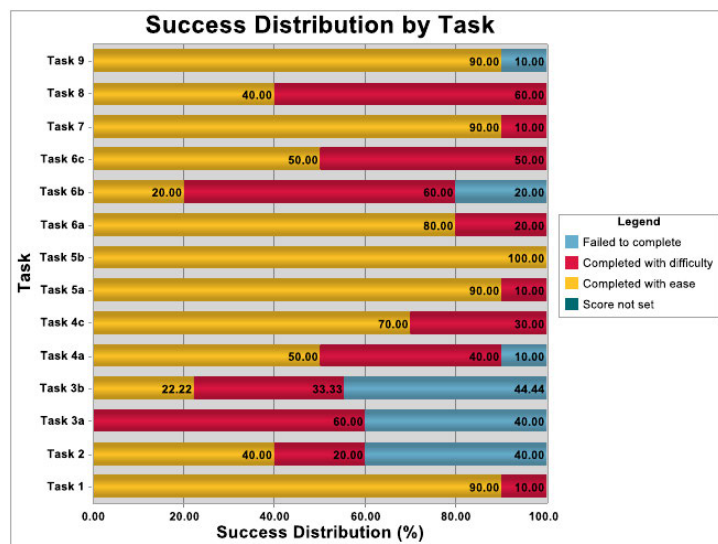


Abbildung 2.2: Dataverse Usability: Erfolgsverteilung bei Szenario 1 [31]

- Das Benutzen der Plattform als jemand, der Daten publiziert (“data creator”). In diesem Szenario mussten die Probanden einen Account und ein Dataverse (Oberverzeichnis eines Projekts in der Dataverse-Software, s. 2.3.2) anlegen, Dateien hochladen, Studien veröffentlichen und sich mit dem Berechtigungssystem auseinandersetzen. [31]

Hierbei kam es zu ähnlichen Ergebnissen: Von den insgesamt elf zu lösenden Aufgaben wurden sieben ($\approx 63,6\%$) von mindestens der Hälfte der “Data Creator” mit Leichtigkeit bewältigt und es wurden wieder alle Aufgaben von mehr Probanden gelöst, als nicht gelöst. [31]

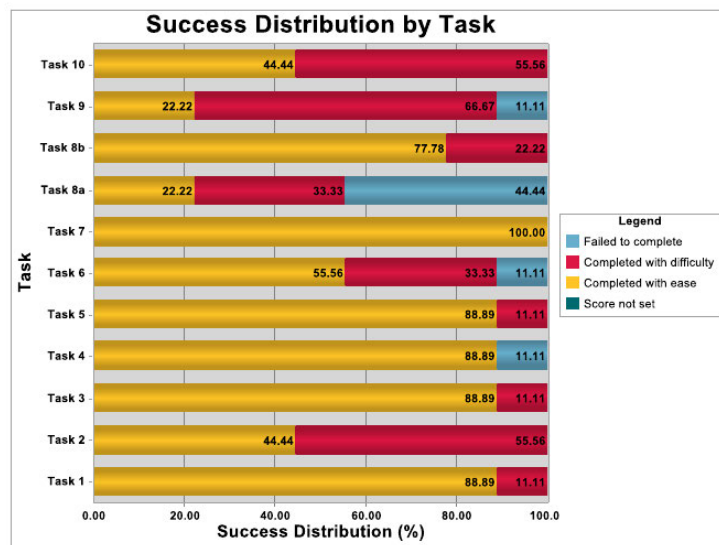


Abbildung 2.3: Dataverse Usability: Erfolgsverteilung bei Szenario 2 [31]

2.3.2 Struktur der Plattform

Innerhalb einer Dataverse-Instanz gibt es ebenfalls vier verschiedene Entitäten: Dataverse, Datasets, User und Groups.

Ein Dataverse ist das Pendant zu einer Organization aus CKAN. Es repräsentiert also eine Institution, die Daten veröffentlichen möchte. Im Gegensatz zu Organizations in CKAN kann ein Dataverse weitere Dataverse beinhalten, sodass hier eine hierarchische Organisationsstruktur möglich ist (z.B. eine Universität mit ihren Fakultäten und die Fakultäten mit ihren Departments), während Organizations in CKAN immer auf einer Stufe stehen.

Datasets funktionieren auf eine ähnliche Weise wie bei CKAN: Sie bestehen aus Titel, beschreibenden Metadaten, aber im Gegensatz zu CKAN nicht aus mindestens einer Datei, sondern können auch leer sein. Datasets können theoretisch auch ohne Zugehörigkeit zu einem Dataverse hochgeladen werden und werden dann automatisch in das Root-Dataverse eingehängt. Außerdem gibt es die Möglichkeit, Dataverses mit anderen Dataverses oder Datasets zu verlinken, womit eine Zusammengehörigkeit zwischen Dataset und Dataverse oder Dataverse und Dataverse dargestellt werden kann.

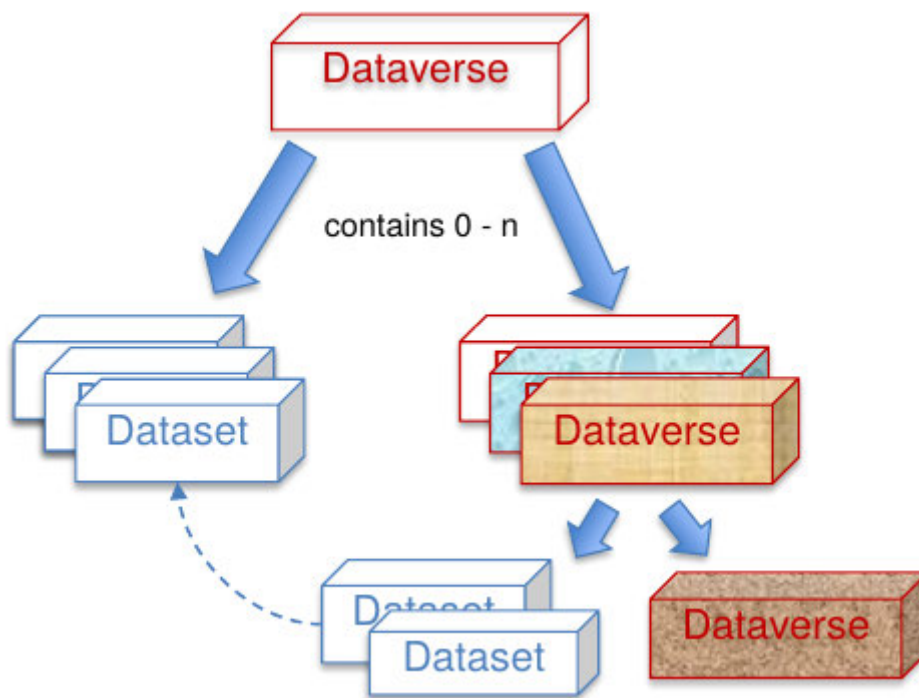


Abbildung 2.4: Dataverse: hierarchische Struktur [29]

Auch Dataverse unterstützt das Hochladen von unbegrenzt großen Dateien unterschiedlichster Formate, aber in der Standardinstallation ist es nicht möglich, den Inhalt der Dateien im Browser in einer Vorschau zu begutachten, ehe man sie herunterlädt.

User können Mitglieder verschiedener Dataverses sein und wie bei CKAN durch ein Berechtigungssystem innerhalb der Dataverses dazu ermächtigt werden, auf Dataverses, Datasets und Dateien zuzugreifen, diese zu editieren und zu löschen.

“Groups” haben bei Dataverse eine andere Bedeutung als bei CKAN. Eine Group ist hier eine Gruppe von Usern. Mithilfe von Groups können schnell Berechtigungen für meh-

rere User und auch für verschiedene Arten von Usern vergeben werden. Zudem können Gruppen neben Usern auch andere Gruppen enthalten, wobei mir bisher noch unklar ist, warum es diese doppelte Möglichkeit zur Berechtigungsverwaltung gibt, dies erscheint redundant.

2.4 Evaluationsmethodik

Miguel et al. [33] haben in ihrer Arbeit mit dem Titel *A Review of Software Quality Models for the Evaluation of Software Products* (2014) eine ganze Reihe von Software-Qualitätsmodellen beschrieben und bewertet. Sie haben die Modelle in drei verschiedene Kategorien unterteilt: “Basic Quality Models”, “Tailored Quality Models” und “Open Source Quality Models”. Unter den Kandidaten befinden sich neben Modellen aus den 1970er-Jahren, wie das McCall-Modell (1977) und das Boehm-Modell (1978) auch aktuellere Modelle wie die ISO 25010 (2008, aktuelle Version 2011) oder das SQO-OSS Quality Model (ebenfalls 2008).

Da die ISO 9126 auf dem McCall- und dem Boehm-Modell basiert [33] und die meisten anderen in der Arbeit vorgestellten Modelle auf der ISO 9126 basieren, fiel die Entscheidung, welches Modell für diese Arbeit angewendet wird, auf dasjenige Modell, welches am besten bewertet wurde: die ISO 25010 - übrigens das offizielle Update der ISO 9126, ein Punkt, der bei der Wahl ebenfalls beeinflussend wirkte.

In Anbetracht dessen, dass die ISO 25010 hauptsächlich eine Ansammlung von Qualitätsmerkmalen und deren Bedeutung ist, wird sie in dieser Arbeit durch die zugehörige ISO 25040 mit dem Titel *Evaluation reference model and guide* ergänzt. Sie beschreibt das Standardvorgehen bei einer Software-Evaluation in fünf Schritten:

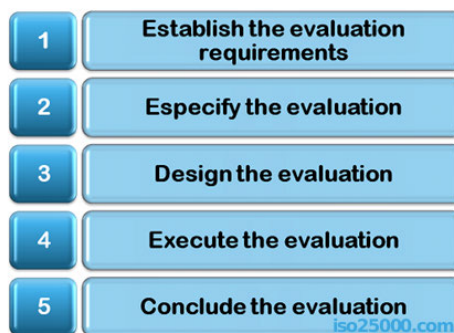


Abbildung 2.5: ISO/IEC 25040: Fünf Schritte einer Software-Evaluation [15]

Im ersten Schritt *Establish the evaluation requirements* werden die Anforderungen der Evaluation festgelegt. Hierzu sollte der Evaluierende sich klar machen und dokumentieren, zu welchem Zweck die Evaluation durchgeführt wird. Danach sollten die Anforderungen an die Softwarequalität anhand eines Qualitätsmodells festgelegt werden. Außerdem wird dokumentiert, welche Teile der Software - falls nicht alle - evaluiert werden sollen. Zuletzt wird die Stringenz der Evaluation definiert. [15]

Im zweiten Schritt *Specify the evaluation* werden die Qualitätsmerkmale, die in die Bewertung einfließen sollen, definiert. Hierzu werden zuerst geeignete Qualitätsmerkmale ausgewählt, dann werden Entscheidungskriterien für die Bewertung der einzelnen Qualitätsmerkmale festgelegt und zum Schluss werden Entscheidungskriterien im Gesamtkontext der Evaluation definiert. [15]

Im dritten Schritt *Design the evaluation* wird ein Ablaufplan für die Evaluation strukturiert. [15]

Der vierte Schritt *Execute the evaluation* ist die Ausführung der Evaluation. Zuerst werden die in Schritt 2 definierten Qualitätsmerkmale gemessen, dann durch die ebenfalls in Schritt 2 definierten Entscheidungskriterien bewertet und zum Schluss in der Gesamtheit betrachtet. [15]

Im fünften und letzten Schritt *Conclude the evaluation* wird die Evaluation abgeschlossen. Zuerst wird das Ergebnis der Evaluation geprüft. Danach wird ein vollständiger Bericht zur Evaluation geschrieben, die Validität der Evaluationsmethode bestätigt und Feedback an den Auftraggeber gegeben. Abschließend wird - je nach Absprache mit dem Auftraggeber - das erhobene Material der Evaluation an den Auftraggeber überstellt, archiviert oder zerstört. [15]

Die ISO/IEC 25010 mit dem Titel *System and software quality models* aus der SQUARE²-Serie der ISO-Standards definiert ein Modell für Softwarequalität bestehend aus acht Kriteriengruppen. Jede Gruppe wiederum besteht aus mehreren einzelnen Kriterien, wie in Abbildung 2.6 zu erkennen ist. Der dritte Schritt (Design der Evaluation) wird mithilfe des zuvor festgelegten Qualitätsmodells ausgearbeitet, während für Schritt vier (Ausführung der Evaluation) eine sequentielle Abarbeitung der zu prüfenden Kriterien geplant ist und für Schritt fünf (Abschluss der Evaluation) keine bestimmte Vorgehensweise avisiert ist, da im Grunde die Ergebnisse nur zusammengefasst werden müssen.

²System and Software Quality Requirements and Evaluation



Abbildung 2.6: ISO/IEC 25010: Gruppierungen von Kriterien [15]

Die dazugehörigen Definitionen sind jeweils kurz und eindeutig definiert und finden sich in Tabelle 2.1 wieder. Auf den ersten Blick und bevor die Evaluation beginnt, wirken alle dargelegten Kriterien sinnvoll und auch im Rahmen dieser Arbeit bewertbar und bearbeitbar. Es gibt jedoch bei genauerem Hinsehen einige geringfügige Mängel, die vor allem damit einhergehen, dass im Rahmen dieser Arbeit der Autor als einziges Versuchssubjekt zur Verfügung steht. Diese Mängel werden in der Fußnote von Tabelle 2.1 erläutert.

1. Functional Suitability	
Attribute	Definition
1.1 <i>Functional Completeness</i>	Degree to which the set of functions covers all the specified tasks and user objectives.
1.2 <i>Functional Correctness</i>	Degree to which a product or system provides the correct results with the needed degree of precision. ¹
1.3 <i>Functional Appropriateness</i>	Degree to which the functions facilitate the accomplishment of specified tasks and objectives.
2. Performance Efficiency	
Attribute	Definition
2.1 <i>Time Behaviour</i>	Degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements.
2.2 <i>Resource Utilization</i>	Degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements.
2.3 <i>Capacity</i>	Degree to which the maximum limits of a product or system parameter meet requirements.
3. Compatibility	
Attribute	Definition
3.1 <i>Co-existence</i>	Degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product.
3.2 <i>Interoperability</i>	Degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.
4. Usability	
Attribute	Definition
4.1 <i>Appropriateness Recognizability</i>	Degree to which users can recognize whether a product or system is appropriate for their needs. ²

2 Grundlagen

4.2 <i>Learnability</i>	Degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use. ²
4.3 <i>Operability</i>	Degree to which a product or system has attributes that make it easy to operate and control.
4.4 <i>User Error Protection</i>	Degree to which a system protects users against making errors.
4.5 <i>User Interface Aesthetics</i>	Degree to which a user interface enables pleasing and satisfying interaction for the user.
4.6 <i>Accessibility</i>	Degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use. ²
5. Reliability	
Attribute	Definition
5.1 <i>Maturity</i>	Degree to which a system, product or component meets needs for reliability under normal operation.
5.2 <i>Availability</i>	Degree to which a system, product or component is operational and accessible when required for use.
5.3 <i>Fault Tolerance</i>	Degree to which a system, product or component operates as intended despite the presence of hardware or software faults.
5.4 <i>Recoverability</i>	Degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system.
6. Security	
Attribute	Definition
6.1 <i>Confidentiality</i>	Degree to which a product or system ensures that data are accessible only to those authorized to have access.
6.2 <i>Integrity</i>	Degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data.
6.3 <i>Non-repudiation</i>	Degree to which actions or events can be proven to have taken place so that the events or actions cannot be repudiated later.
6.4 <i>Authenticity</i>	Degree to which the identity of a subject or resource can be proved to be the one claimed.
6.5 <i>Accountability</i>	Degree to which the actions of an entity can be traced uniquely to the entity.
7. Maintainability	
Attribute	Definition
7.1 <i>Modularity</i>	Degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components.
7.2 <i>Reusability</i>	Degree to which an asset can be used in more than one system, or in building other assets.
7.3 <i>Analysability</i>	Degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified.
7.4 <i>Modifiability</i>	Degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality.
7.5 <i>Testability</i>	Degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met.
8. Portability	
Attribute	Definition
8.1 <i>Adaptability</i>	Degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments.
8.2 <i>Installability</i>	Degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment.
8.3 <i>Replaceability</i>	Degree to which a product can replace another specified software product for the same purpose in the same environment.

1: Dieses Kriterium ist besser für die Bewertung eines im Rahmen einer Auftragsarbeit implementierten Systems geeignet, als für eine Evaluation zweier Open-Source-Projekte.

2: Da der Autor als einziges Versuchssubjekt dieser Arbeit dient und er jahrzehntelange Erfahrung im Umgang mit verschiedensten Softwaresystemen hat, ist dieses Kriterium wenig aussagekräftig.

Tabelle 2.1: ISO/IEC 25010: Qualitätsmerkmale und deren Definitionen [15]

Laut Holz auf der Heide [32] kann die Art einer Evaluation anhand dreier Eigenschaften bestimmt werden: die Ziele, die Kriterien und die Mittel ebendieser Evaluation. Die drei dazugehörigen Fragen “Wozu, Was und Wie soll evaluiert werden?” sollen bei der Charakterisierung der Evaluation helfen. Bei einer vergleichenden Evaluation werden zwei oder mehr verschiedene Systeme dahingehend überprüft, welches der Systeme für einen Anwendungsfall besser geeignet ist. Zunächst wird überprüft, inwiefern die Anforderungen für die beiden Anwendungsfälle bereits durch die gegebenen Funktionalitäten der beiden getesteten Systeme abgedeckt sind.

Holz auf der Heide [32] führte 1993 weiterhin aus, dass bei einer vergleichenden Evaluation hauptsächlich die Frage “Which is better?” beantwortet werden soll. Hierzu böten sich zwei Arten von Kriterien besonders gut an: die subjektive Zufriedenheit des Benutzers sowie objektiv messbare Merkmale wie *Fehler und Bearbeitungszeiten* der Benutzer. Durch letzteres Kriterium ließe sich die intuitive Bedienbarkeit der beiden Systeme messen.

Vergleicht man diese Aussage mit der Abbildung und der Tabelle aus der ISO 25010, sieht man, dass in den knapp zwanzig Jahren, die zwischen beiden Veröffentlichungen liegen, nicht übermäßig viele neue objektive Kriterien für die Evaluation von Software entstanden sind. Vielmehr sind tatsächlich ein großer Teil der Kriterien subjektiver Natur.

3 Analyse

In diesem Kapitel werden die Anforderungen, die die beiden Anwendungsfälle an die beiden Systeme stellen, erhoben. Hierzu werden die Ausgangssituationen beider Anwendungsfälle analysiert. Außerdem werden Voraussetzungen für die Bewertung der Systeme festgelegt und offensichtliche Defizite bezüglich der formulierten Anforderungen festgestellt. Zudem wird darauf eingegangen, wie diese Defizite durch eine Erweiterung der Systeme gegebenenfalls ausgeglichen werden können.

3.1 Anforderungen an die Systeme

Im ersten der drei folgenden Unterabschnitte wird dargelegt, in welcher Ausgangssituation sich das Big Data Labor der HAW Hamburg befindet und welche Anforderungen es an die zu bewertenden Systeme stellt. Der zweite Unterabschnitt behandelt denselben Inhalt für die Hamburg City Health Study, während im dritten Unterabschnitt Voraussetzungen für eine erfolgreiche Bewertung der Systeme definiert werden.

3.1.1 Situation im HAW Big Data Labor

Laut Prof. Dr. Olaf Zukunft werden an der HAW Hamburg auf drei verschiedene Arten von Forschenden Daten produziert (persönliche Kommunikation, 07. April 2020). Zum Einen werden Untersuchungen von Mitarbeitern durchgeführt, zum Anderen schreiben Studierende ihre Abschlussarbeiten und auch in verschiedenen Lehrveranstaltungen entstehen durch die gemeinsame Arbeit von Professoren, wissenschaftlichen Mitarbeitern und Studenten Daten. Diese Daten sind in jeglicher Hinsicht heterogener Natur: sie dienen unterschiedlichsten Zwecken, sind oft sehr spezifisch und kommen in unterschiedlichsten Mengen, Größen und Dateiformaten vor. Neben "normalen" Datenformaten wie CSV und XML gibt es beispielsweise Eyetracking-Daten, die aus mehreren Gigabyte großen Filmdateien bestehen, oder pseudonymisierte Versicherungsdaten, oder Daten, die aus

juristischer Sicht nicht für jeden Nutzer zugänglich sein dürfen.

Eine weitere Besonderheit der Situation der HAW ist, dass das Personal und die Studierenden typischerweise nach der Fertigstellung ihrer Arbeit - sei es drittmittelfinanzierte Forschung oder Abschlussarbeit - die HAW verlässt und die Daten nicht mehr von ihnen verwendet werden.

Die aktuelle Infrastruktur des Departments Informatik der HAW bietet neben der Arbeit an privaten Rechnern vier verschiedene Möglichkeiten, praktische Arbeiten für die o.g. Tätigkeiten durchzuführen:

1. virtuelle Maschinen, die sowohl von Mitarbeitern als auch von Studenten für ihre Arbeiten genutzt werden;
2. das Big Data Labor mit 16 Rechnern und sehr viel Speicherplatz, welches ebenfalls von Studenten für Experimente genutzt wird;
3. die ICC (Informatik Compute Cloud), eine containergestützte Cloud-Umgebung in der Applikationen in Form von Docker Containern gehostet werden können;
4. private Cloud-Dienste wie AWS (Amazon Web Services), GCP (Google Cloud Platform) und Microsoft Azure.

Neben der Heterogenität der Daten gibt es also auch eine Heterogenität bei der Auswahl der Speichermedien. Besser wäre es, wenn all diese Daten inklusive schriftlicher Ergebnisse der Arbeiten nach Abschluss des Projekts an einem zentralen Ort gespeichert werden würden. So wäre es für zukünftige Angestellte und Abschlussarbeitschreibende einfacher, an die Arbeiten ihrer Vorgänger anzuknüpfen und die wertvolle Zeit, die für die Forschungsarbeit aufgewendet wurde und wird, würde noch effizienter genutzt werden. Zusammengefasst sind die Anforderungen des HAW Big Data Labors an die beiden Systeme also

- die Unterstützung verschiedenster Datentypen,
- ein Berechtigungssystem für den Zugriff auf Daten,
- eine gebündelte Darstellung von Daten und (schriftlichen) Ergebnissen.

3.1.2 Situation der Hamburg City Health Study

Die Hamburg City Health Study (HCHS) ist mit geplanten 45.000 Probanden die größte lokale Gesundheitsstudie der Welt und wird vom Universitätsklinikum Hamburg-Eppendorf (UKE) durchgeführt. [13] Die Erfassung, Qualitätssicherung und Bereitstellung einer solch großen Menge von Daten gestaltet sich schwierig, da die Daten in unterschiedlichsten Formaten und Mengen über verschiedene Übertragungswege gesammelt und weiterverarbeitet werden müssen, ehe sie zu Forschungszwecken verwendet werden können. Durch verschiedene Hürden, wie zum Beispiel die EU-DSGVO, ist die Suche nach einem geeigneten Datenkatalog-Tool zur Beschreibung der vorhandenen Daten sowie Bereitstellung dieser Daten für die Forscher der über 30 an der Studie beteiligten Kliniken des UKE ebenso schwierig, auch weil die Anforderungen der HCHS durch die Verknüpfung kritischer medizinischer sowie persönlicher Daten sehr speziell sind.

Das System muss vollständig eigenständig arbeiten und darf nicht mit außenstehende Applikationen kommunizieren, die abseits der UKE-eigenen Hardware gehostet sind. Der Katalog der zur Verfügung stehenden Daten soll einsehbar sein, bevor die Daten tatsächlich in Gänze verfügbar sind, damit forschungswillige Wissenschaftler ihre Forschungsanträge schon rechtzeitig ausarbeiten, vorstellen und genehmigen lassen können, bevor sie Zugriff auf die Daten erhalten und ihre Forschungsarbeiten und -daten in dem Tool aufbereiten und zur Verfügung stellen. Die gewünschten Daten werden in den Forschungsanträgen nicht vage, wie etwa “alle Echokardiographie-Daten”, sondern auf die einzelne Variable genau angegeben, um dem Prinzip der Datensparsamkeit gerecht zu werden. Also sollten diese auch im Katalog ebenso genau angegeben werden können. Zudem soll nicht jeder Wissenschaftler auf alle Daten zugreifen können, sondern jeder nur auf die in seinem bewilligten Forschungsantrag angegebenen Daten.

Noch einmal zusammengefasst sind die Anforderungen der HCHS an die beiden zu bewertenden Tools also neben den trivialen Grundfunktionen wie Up-, Download und Veröffentlichung von Daten folgende:

- vollständige Funktionsfähigkeit ohne Abhängigkeiten von Fremdsystemen,
- Datenschutzkonformität,
- ein ausgefeiltes Berechtigungssystem,
- ein von den einzelnen Datensätzen unabhängiges, individualisierbares Katalogsystem.

3.1.3 Voraussetzungen für die Bewertung der Systeme

Einige der in Tabelle 2.1 festgehaltenen Kriterien zur Bewertung eines Systems erfordern eine genaue Definitionen dessen, was von dem System erwartet wird. Im Folgenden finden sich für alle Kriterien, die einen definierten Rahmen benötigen, die notwendigen Definitionen:

1. **Functional Suitability:** Für die erste Kriteriengruppe müssen die Aufgaben und Ziele der Benutzer festgelegt werden. Viele der grundlegenden Aufgaben und Ziele der beiden Anwendungsfälle “HAW Big Data Labor” und “Hamburg City Health Study” überschneiden sich, daher werden sie in der nachfolgenden Tabelle gemeinsam aufgelistet.

Ziele	
Kennung	Ziel
Z01	Abbildung der Organisationshierarchie im Berechtigungssystem
Z02	Zugriff auf bestimmte Daten/Dateien/Projekte nur durch bestimmte Nutzer
Z03	geordnete Sammlung aller erhobenen Forschungsdaten
Z04	Kategorisierung von Projekten (Tagging)
Z05	Zuordnung von schriftlichen Ergebnissen zu Daten und Dateien
Z06	Teilen von Forschungsdaten mit anderen Forschern
Z07	Abbildung eines Variablenkatalogs unabhängig von tatsächlich hochgeladenen Dateien (nur HCHS)
Aufgaben	
Kennung	Aufgabe
A01	Upload vieler Dateien in einem Projekt
A02	Upload von Dateien bis 10GB Größe
A03	Suchen und Finden von Dateien
A04	Suchen und Finden von Projekten
A05	Suchen und Finden von Benutzergruppierungen
A06	Suchen und Finden von einzelnen Benutzern
A07	Download von Dateien <i>mit entsprechender Autorisierung</i>
A08	Verwalten von einzelnen Benutzern
A09	Verwalten von Benutzern in Gruppen
A10	Veröffentlichen von Forschungsergebnissen
A11	Bearbeiten des Online-Variablenkatalogs im Browser (nur HCHS)

Tabelle 3.1: Für die Kategorie Functional Suitability benötigte Ziele und Aufgaben

2. **Performance Efficiency:** Für das erste Kriterium dieser Gruppe, *Time Behaviour*, sollte festgelegt werden, wie lange das System für das Verarbeiten von Arbeitsschritten benötigen darf. Laut Ahmed et al. [28], die in ihrer Arbeit APM¹-Tools getestet haben, benutzt zum Beispiel das APM-Tool “Pinpoint” eine Sekunde als Grenzwert für eine gute Bewertung der Antwortzeit eines Systems.

Auch auf DNSStuff.com, einer Review-Website, die zu der amerikanischen Software-Firma Solarwinds gehört, findet sich in einem Artikel über Application Response Time Monitoring eine Einschätzung zu dem Thema. Eine Antwortzeit von 0,1 Sekunde sei für einen Benutzer eine nicht zu merkende Unterbrechung. Eine Sekunde sei die maximal zu akzeptierende Antwortzeit, da der Benutzer die Verzögerung kaum merke. Alles über einer Sekunde sei jedoch problematisch und bei fünf bis sechs Sekunden beendeten bzw. verließen die Benutzer typischerweise das Programm oder die Website vollständig. [3]

Abgeleitet aus den Erkenntnissen der beiden letzten Absätze sowie persönlicher Eindrücke beim Arbeiten mit verschiedenster Software, scheint der 1-Sekunden-Benchmark als Grenze für eine gute Bewertung der beiden Systeme ebenfalls angemessen.

Das zweite Kriterium, *Resource Utilization* benötigt festgelegte Hardware-Ressourcen. Da im Rahmen dieser Arbeit virtuelle Maschinen mit 4GB RAM und 2 Prozessorkernen zum Einsatz kommen, wird mit dieser Hardware-Spezifikation gemessen.

3. **Compatibility:** Für die beiden Kriterien dieser Kategorie sind keine für die Anwendungsfälle spezifischen Definitionen erforderlich.
4. **Usability:** Um die Systeme hinsichtlich des Kriteriums *User Interface Aesthetics* bewerten zu können, muss vorher klar sein, wodurch sich ein angenehmes und befriedigendes Erlebnis bezüglich der Ästhetik eines Systems beim Arbeiten auszeichnet. Eine angenehme Farbwahl für die Benutzeroberfläche sowie ein hoher Kontrast zwischen Hintergrund- und Schriftfarbe als auch eine gut gewählte Schriftgröße ist für längerfristige Arbeiten von großer Bedeutung, damit auch die unterschiedlich guten Augen verschiedener Mitarbeiter mit der Oberfläche zurechtkommen. Zudem sollten Schaltflächen eindeutig beschriftet sein, sodass jeder Mitarbeiter intuitiv abschätzen kann, welche Konsequenzen mit welcher Aktion einhergehen. In Kombination schaffen diese Eigenschaften einen Grad von Wohlbefinden beim Umgang

¹Application Performance Measurement

mit dem System. Die anderen Kriterien dieser Gruppe benötigen keine weiteren Definitionen.

5. **Reliability:** Für die Kriterien dieser Kategorie sind keine für die Anwendungsfälle spezifischen Definitionen erforderlich.
6. **Security:** Für die Kriterien dieser Kategorie sind keine für die Anwendungsfälle spezifischen Definitionen erforderlich.
7. **Maintainability:** Für die Kriterien dieser Kategorie sind keine für die Anwendungsfälle spezifischen Definitionen erforderlich.
8. **Portability:** Für die Kriterien dieser Kategorie sind keine für die Anwendungsfälle spezifischen Definitionen erforderlich.

3.2 Identifikation von Defiziten bzgl. der Anforderungen

Im folgenden Abschnitt wird beschrieben, welche der Anforderungen von den jeweiligen Systemen nicht oder nicht gut erfüllt werden. Während die in 3.1.1 beschriebenen Anforderungen des HAW Big Data Labors nach einer Handvoll Stunden Auseinandersetzung mit beiden Systemen von beiden vollständig erfüllt zu werden scheinen, sieht es durch die spezielleren Anforderungen der HCHS für diesen Anwendungsfall etwas schlechter aus.

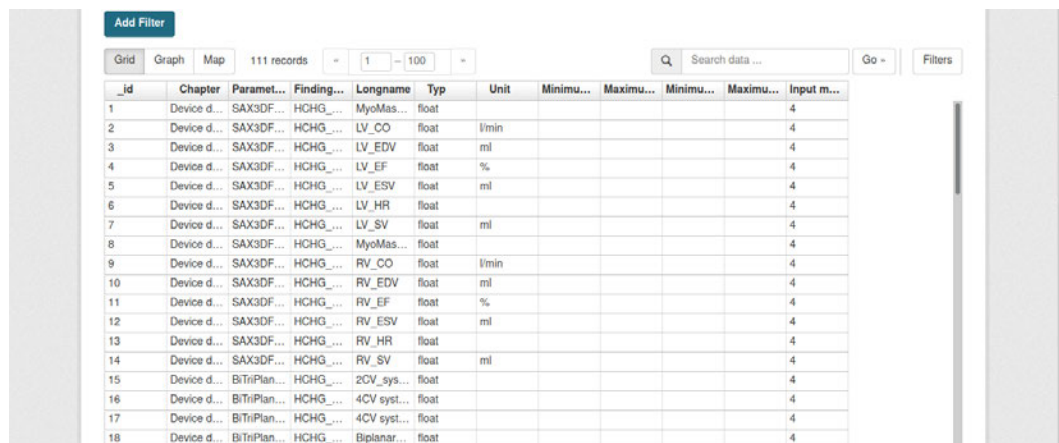
Keines der beiden untersuchten Systeme weist eine eingebaute Funktionalität auf, die es ermöglicht einen Variablenkatalog unabhängig von bereits hochgeladenen Dateien zur Verfügung zu stellen. Zudem möchte Dataverse beim Veröffentlichen von Datensätzen mit "DataCite" kommunizieren. DataCite ist eine Plattform, bei der der DOI (Digital Object Identifier) des zu veröffentlichenden Datensatzes hinterlegt werden kann. Der Ansatz ist im Grunde gut, verständlich und unterstützt den Open-Science-Gedanken, unter dessen Prämisse das Dataverse-Projekt hervorgegangen ist, verträgt sich aber leider nicht mit der Anforderung der HCHS, dass das System nicht mit fremdgehosteten Applikationen kommunizieren soll.

3.3 Beheben der Defizite durch Implementation von Extensions

Das Defizit, welches ausschließlich Dataverse und die HCHS betrifft, also die Kommunikation mit einem Fremdsystem, lässt sich leider nicht so leicht umgehen oder mithilfe einer Code-Erweiterung oder -Veränderung lösen. Dies gilt aber nicht für den von bereits hochgeladenen Dateien unabhängigen Variablenkatalog. Dieser Variablenkatalog enthält letztlich Variablennamen (in der Regel Abkürzungen), Beschreibungen (in der Regel ein bis einige Worte) und Informationen über Einheiten oder Datentypen in einer tabellarischen Form.

3.3.1 Variablenkatalog für CKAN

In CKAN ist es von Haus aus möglich, tabellarische Dateiformate wie CSV und TSV zu speichern und mithilfe des sogenannten *Data Explorers* direkt im Browser einzusehen. Für Excel-Dateien (xlsx) wird in der Dokumentation auch geworben, hier funktioniert in der für diese Arbeit angelegten CKAN-Instanz (Version 2.8 in Docker) die Preview im Browser nur teilweise. Visuell würde die Darstellung im Data Explorer ausreichen:



_id	Chapter	Paramet...	Finding...	Longname	Typ	Unit	Minimu...	Maximu...	Minimu...	Maximu...	Input m...
1	Device d...	SAX3DF...	HCHG_...	MyoMas...	float						4
2	Device d...	SAX3DF...	HCHG_...	LV_CO	float	l/min					4
3	Device d...	SAX3DF...	HCHG_...	LV_EDV	float	ml					4
4	Device d...	SAX3DF...	HCHG_...	LV_EF	float	%					4
5	Device d...	SAX3DF...	HCHG_...	LV_ESV	float	ml					4
6	Device d...	SAX3DF...	HCHG_...	LV_HR	float						4
7	Device d...	SAX3DF...	HCHG_...	LV_SV	float	ml					4
8	Device d...	SAX3DF...	HCHG_...	MyoMas...	float						4
9	Device d...	SAX3DF...	HCHG_...	RV_CO	float	l/min					4
10	Device d...	SAX3DF...	HCHG_...	RV_EDV	float	ml					4
11	Device d...	SAX3DF...	HCHG_...	RV_EF	float	%					4
12	Device d...	SAX3DF...	HCHG_...	RV_ESV	float	ml					4
13	Device d...	SAX3DF...	HCHG_...	RV_HR	float						4
14	Device d...	SAX3DF...	HCHG_...	RV_SV	float	ml					4
15	Device d...	BiTriPlan...	HCHG_...	2CV_sys...	float						4
16	Device d...	BiTriPlan...	HCHG_...	4CV_syst...	float						4
17	Device d...	BiTriPlan...	HCHG_...	4CV_syst...	float						4
18	Device d...	BiTriPlan...	HCHG_...	Biplanar...	float						4

Abbildung 3.1: CKAN: simple und übersichtliche Darstellung von Tabellendaten

Der Data Explorer bietet eine Suchfunktion sowie die Möglichkeit zum Filtern nach bestimmten Werten in den einzelnen Tabellenspalten an. Diese Funktionen sind zwar

hilfreich, aber leider gibt es einige Probleme, die zumindest eine Erweiterung der Data Explorer-View erfordern:

1. Die Struktur von CKAN, wie in 2.2.2 beschrieben, limitiert die Verwendung des Data Explorers auf hochgeladene Tabellendateien. Das Hochladen wiederum ist dahingehend limitiert, dass Dateien nur projektgebunden (also in Datasets, die einer Organization gehören) hochgeladen werden können - was aufgrund der Systemstruktur Sinn ergibt - aber gegen die Anforderung verstößt, dass der Katalog unabhängig von bereitgestellten Daten abrufbar sein soll. Man könnte nun ein Projekt mit dem Namen "Variablenkatalog" anlegen und dort entweder eine Tabellendatei pro Untersuchung hochladen, was viel Klicken nach sich zöge und dadurch nicht besonders ergonomisch wäre, oder eine große Datei mit allen Variablen die während der Studie erhoben werden, aber diese Lösungen sind dilettantisch, weil beide Ansätze das Dataset zweckentfremden.
2. Ebenso kritisch ist, dass die hochgeladenen Tabellendateien nicht im Browser bearbeitet werden können. Um Änderungen vorzunehmen, müsste also die Datei heruntergeladen, in Microsoft Excel o.Ä. bearbeitet, erneut hochgeladen und die Vorgängerdatei deaktiviert bzw. gelöscht werden.

Es wäre schöner und besser gelöst, wenn der Variablenkatalog nicht mithilfe von Tabellendateien, sondern als Tabelle(n) in der PostgreSQL-Datenbank des Systems gespeichert wäre. In der Dokumentation von CKAN gibt es neben einem API Guide einen umfangreichen Abschnitt über das Erweitern des Systems mittels eigener Python-Skripte (Extension guide). Hiermit könnte - eventuell aufbauend auf der Data Explorer-View - eine neue View konzipiert und implementiert werden, welche in der Menüleiste der Webapplikation aufrufbar ist. Diese View könnte neben den bereits gegebenen Funktionen "Suchen" und "Filtern" auch noch die Funktion "Bearbeiten" erfüllen - natürlich nur für bestimmte Benutzer, für diesen Anwendungsfall speziell zum Beispiel Mitarbeiter des Datenmanagements der Studie.

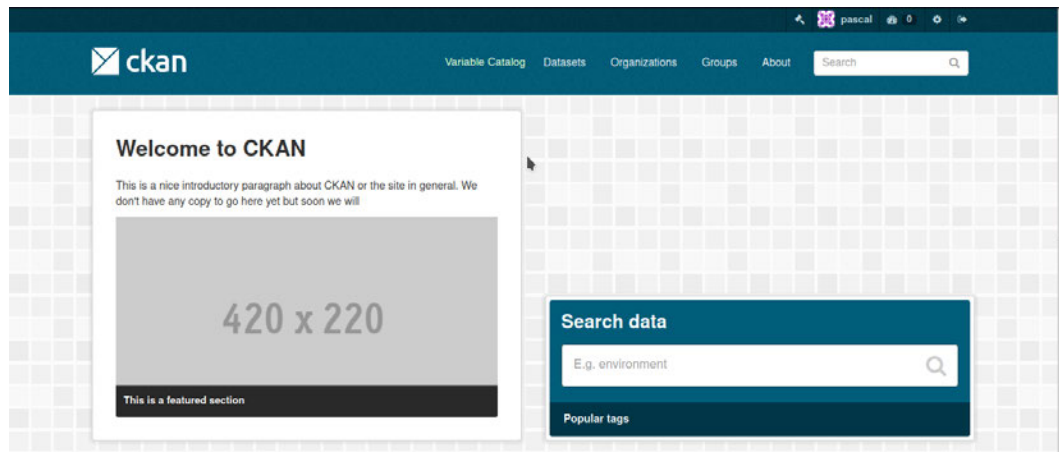


Abbildung 3.2: CKAN: Mögliche Platzierung des Variablenkatalog-Buttons

Falls die Wiederverwendung des Data Explorers sich aus irgendeinem Grund als schlechte Idee herausstellen sollte, ist glücklicherweise bekannt, welche Library für ihn verwendet wurde: Recline.js. Diese Library wurde von der Open Knowledge Foundation (OKFN) entwickelt, welche auch an CKAN maßgeblich beteiligt ist. Der Dokumentation sowie der Demo auf der Website von Recline.js ist zu entnehmen, dass mit dieser Library nicht nur Daten präsentiert (wie im Data Explorer von CKAN), sondern auch bearbeitet werden können. [14] Damit wäre die Anforderung erfüllt.

3.3.2 Variablenkatalog für Dataverse

Im Gegensatz zur Dokumentation von CKAN existiert in der Dokumentation von Dataverse kein Abschnitt über das Erstellen eigener Extensions. Für jeden, der nicht die Absicht hat, sich mit dem gesamten Source Code auseinanderzusetzen - der durchaus vollständig einseh- und veränderbar ist - um das System an die eigenen Anforderungen anzupassen, existiert in der Dokumentation allerdings ein recht umfassender API Guide, in dem sich unter anderem der Abschnitt "Building External Tools" wiederfindet. Dieser Abschnitt schlägt vor, dass nicht der Code von Dataverse verändert wird, sondern ein neues, eigenes Tool mittels der API auf Dataverse-Daten zugreift. Diese externen Tools können dann mittels eines "Explore"- oder "Configure"-Buttons *auf der Landing-Page eines Datasets oder einer Datei* eingebunden werden. Das ist leider genau das gleiche Problem, welches bei CKAN unter anderem dazu führt, dass der Data Explorer nicht

ausreicht; das externe Tool bezöge sich auch hier auf die einzelnen Datasets.

Auch hier scheint es eine Möglichkeit zu geben, das Problem zu umgehen: Der Variablenkatalog wird wie auch bei CKAN in der PostgreSQL-Datenbank des Systems gespeichert. Mittels API - oder direktem Zugriff auf die Datenbank, da ein API-Call keinen Mehrwert zu bieten scheint - ruft das externe Tool den Katalog ab und stellt ihn auf einer in diesem Fall externen, also nicht zum Dataverse selbst gehörenden Website tabellarisch dar. Falls sich Recline.js als gut handhabbar herausstellen sollte, spräche nichts gegen die erneute Verwendung dieser Library für den Variablenkatalog im Dataverse.

Um sicherzustellen, dass Benutzer des Dataverses den Variablenkatalog auch finden, lässt sich womöglich ebenfalls ein Button in der Menüleiste integrieren, ansonsten wäre eine Verlinkung in der Beschreibung des Root-Dataverses ebenfalls denkbar.

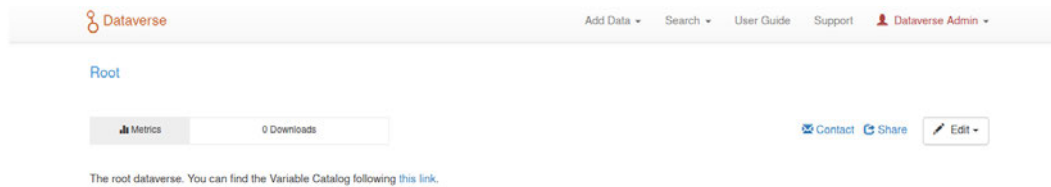


Abbildung 3.3: Dataverse: Verlinkung des Variablenkatalogs im Root-Dataverse

4 Entwurf einer Extension

Im Folgenden wird der Entwurf einer Extension für CKAN beschrieben. Wie in 3.3 bereits erwähnt wird, ist das Ziel der zu entwerfenden Extension die Darstellung sowie das Schaffen einer Möglichkeit zur Bearbeitung eines tabellarischen Variablenkatalogs im Browser. Dieser Katalog soll unabhängig von tatsächlich bereits hochgeladenen Datasets existieren, um somit einen Überblick über die zur Forschung verwendbaren Daten zu schaffen.

Da das Entwickeln von CKAN-Extensions in einer Docker-Umgebung auf einem virtuellen Server durch das Fehlen einer IDE¹ und Arbeiten in der Konsole sehr unkomfortabel ist, wurde für den Zweck der Entwicklung eine “native” CKAN-Instanz auf Ubuntu 16.04 installiert. Die erste Idee ließ sich nach etwas genauerer Recherche im CKAN-Quellcode nicht so gut umsetzen wie erhofft, daher ist die Herangehensweise an dieses Teilprojekt der Arbeit iterativer Natur. Insgesamt erfolgten in der Konzeptionsphase drei Iterationen, die inkrementell mehr Erfolg mit sich brachten.

4.1 Iteration 1: Wiederverwendung des DataExplorers

Die Wiederverwendung des DataExplorers, einer View, mithilfe derer in CKAN Tabledateien aus Datasets in der Weboberfläche angezeigt werden können (s. Abb. 3.1) erscheint auf den ersten Blick logisch, denn zwei der gewünschten Funktionalitäten (Darstellung einer Tabelle, Suchen und Filtern in der Tabelle) sind darin bereits enthalten. Zudem bietet das darin verwendete Framework namens “Recline.js” die Möglichkeit, die dargestellte Tabelle online zu bearbeiten. [18] Idealerweise hätte die Datenquelle auf eine relationale Datenbank - und damit auf eine persistenterere Variante der Datenspeicherung - geändert werden können, sodass sich das folgende lose an das MVC²-Muster angelehnte Schema ergeben hätte:

¹Integrated Development Environment

²Model-View-Controller

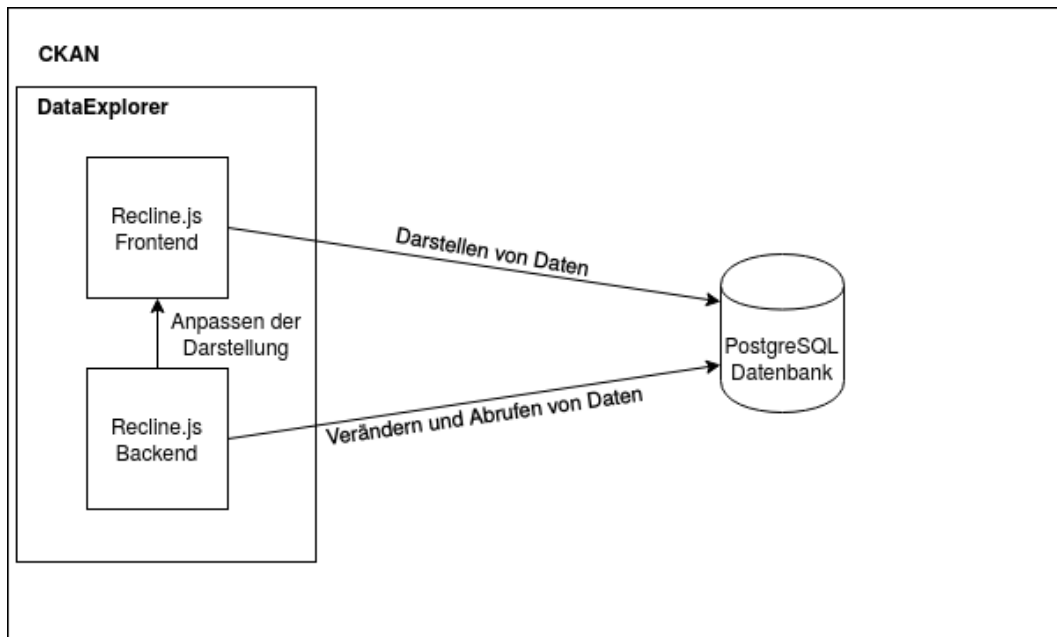


Abbildung 4.1: CKAN-Extension: MVC-ähnliches Schema

Wie sich bei genauerem Untersuchen der Dokumentation von Recline.js herausstellt, gibt es zwar für einige Datenquellen-Typen wie z.B. CSV, Excel oder auch CouchDB sogenannte Backends, die die anzuzeigenden Daten aus der entsprechenden Datenquelle abrufen und speichern können, aber unter den unterstützten Datenquellen findet sich kein Backend für ein relationales Datenbanksystem.[20]

In Anbetracht des avisierten Zeitaufwands für die Entwicklung der Extension kommt ein selbst implementiertes Backend für relationale Datenbanken nicht infrage. Daher wird in dieser ersten Iteration auf das Implementieren verzichtet, auch weil ein weiteres infrage kommendes Framework namens "Tabulator" mit dem Laden von Daten mittels AJAX-Request und JSON auf sich aufmerksam machte, wie in 4.2 noch genauer erläutert wird.

4.2 Iteration 2: Das Framework Tabulator

Da die Nutzung des CKAN DataExplorers an der fehlenden Unterstützung für relationale Datenbanken als Datenquelle gescheitert ist, wurde bei der Auswahl des Frameworks Tabulator, welches dieselben Funktionalitäten bzgl. der Darstellung und Manipulation von Tabellendaten im Browser bietet, darauf geachtet, dass dieses Problem nicht besteht.

Laut Dokumentation ist es mit Tabulator möglich, mittels AJAX-Requests und JSON Daten zu laden und im Tabulator-Frontend darstellen zu lassen. [27]

Die daraus resultierenden Änderungen am Konzept sind in der nachfolgenden Abbildung zu sehen:

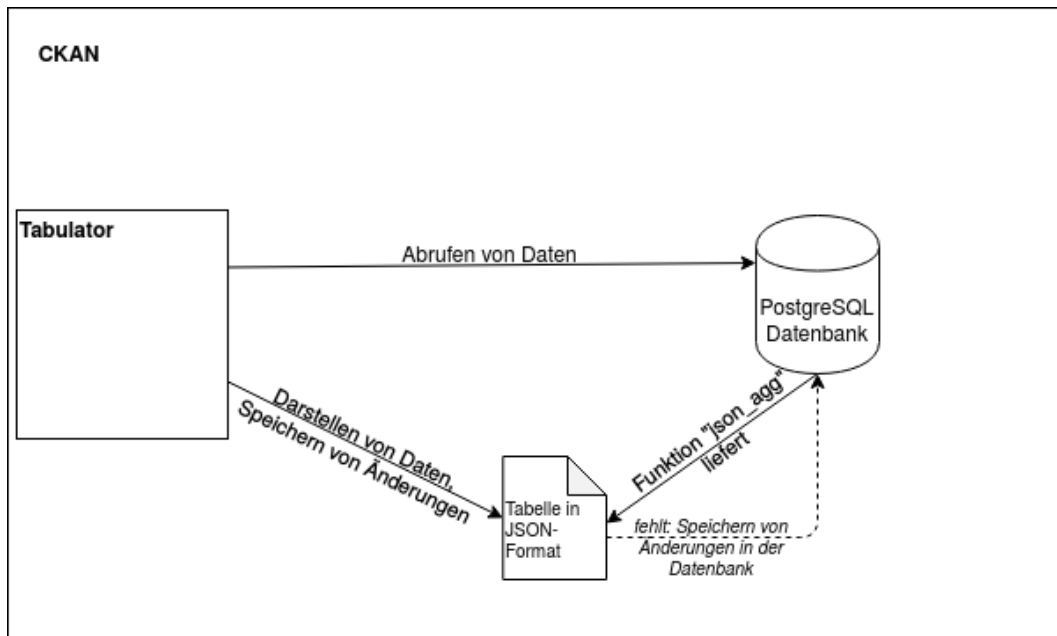


Abbildung 4.2: CKAN-Extension: Entwurf mit Tabulator

In 5.2 wird darauf eingegangen, wieso die Implementation trotz des vielversprechenden ersten Eindrucks, den die Dokumentation von Tabulator durch versprochene Features vermittelt, nicht erfolgreich verlaufen ist.

4.3 Iteration 3: Verwendung von “rohem” Recline.js mit CSV

Die Rückkehr zu einer abgewandelten Version der ursprünglichen Idee, in der die relationale Datenbank als Datenquelle durch eine auf dem Webserver abgespeicherte CSV-Datei ersetzt wird, erfolgt aufgrund mehrerer Probleme mit dem Framework Tabulator, wie in 5.2 genauer beschrieben wird. Das Konzept hat sich dadurch verglichen mit Iteration 1 nicht viel verändert: die Datenquelle ist nun eine andere, da das gewünschte Backend

existiert und es wird nicht die in CKAN schon verwendete, angepasste Version von Recline.js wiederverwendet, sondern das Framework in reiner Form für den eigenen Zweck benutzt.

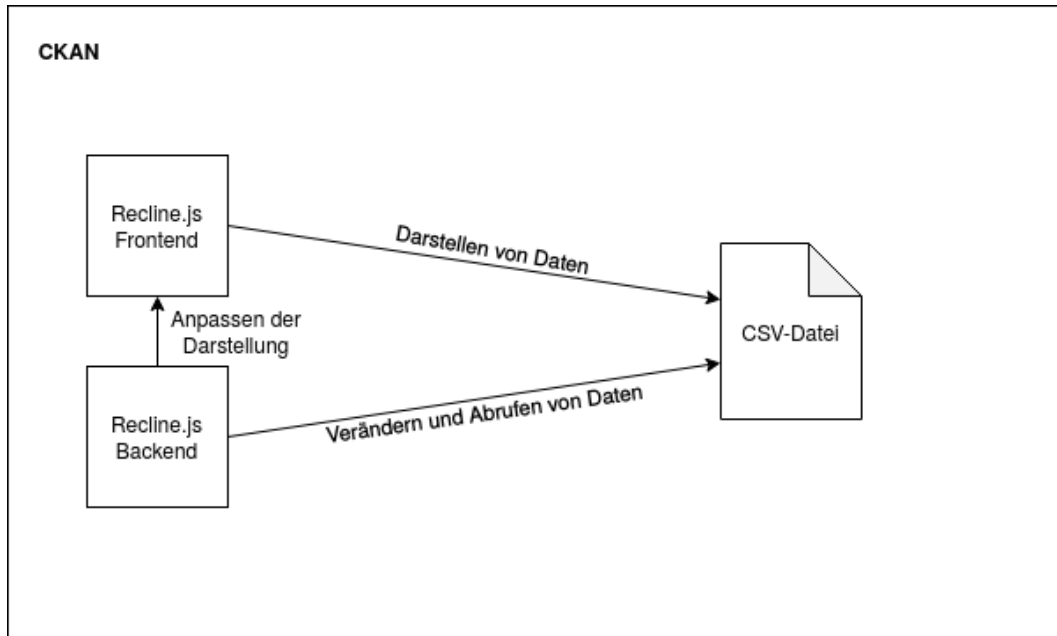


Abbildung 4.3: CKAN-Extension: Entwurf mit Recline.js

In 5.3 wird erläutert, welche Schwierigkeiten zu meistern waren und wie die Probleme gelöst werden konnten.

5 Implementation einer Extension

In diesem Kapitel wird erläutert, wie an die Implementation der Extension für CKAN herangegangen wurde. In den Abschnitten 3.2 und 3.3 wurde ein zentrales Defizit identifiziert und grob beschrieben, wie es behoben werden kann. Im Kapitel 4 wird in drei Iterationen ein Konzept erarbeitet, während in diesem Kapitel jeweils zu den drei Iterationen beschrieben wird, welche Auffälligkeiten bei der Implementierung des Konzepts aufgetreten sind. Dabei wurde jeweils zuerst versucht, der Javascript- und HTML-Code zum Laufen zu bringen und den Code in die Applikation zu integrieren.

5.1 Iteration 1: Wiederverwendung des DataExplorers

Wie in 4.1 bereits erwähnt, wurde in Anbetracht dessen, dass mit dem Framework “Tabulator” eine augenscheinlich bessere Alternative existiert, davon abgesehen, einen Implementierungsversuch durchzuführen. Abgesehen von dem nicht existierenden “Backend” für relationale Datenbanken als Datenquelle ist das darunterliegende Framework “Recline.js” auf eine Art und Weise verwendet worden, die die Funktionalität in einem Maße restriktiert, dass es simpler wäre, Recline.js selbst zu verwenden.

5.2 Iteration 2: Das Framework Tabulator

Da in dem im Versuch verwendeten Datenbanksystem PostgreSQL eine Funktion namens “json_agg” (= JSON aggregate) existiert, die eine Tabelle ins JSON-Format bringt und ausgibt und das Tabulator-Framework laut Dokumentation JSON-Dateien auf sehr simple Weise laden und darstellen kann [27], fiel die Wahl für die Datenübertragung aus der Datenbank ins Tabellen-Frontend auf JSON. Was allerdings beim Durchlesen der Dokumentation von Tabulator nicht auffiel, ist dass hierbei keine “echte” JSON-Notation

verwendet wird, sondern Tabulator mit der sehr ähnlichen, aber eben nicht gleichen Notation eines Javascript-Arrays arbeitet. In einer JSON-Datei müssen sowohl der Schlüssel als auch der Wert in doppelte Anführungszeichen gesetzt werden, während ein JavaScript-Array nur String-Werte in Anführungszeichen setzt. Retrospektiv hätte dieses Problem sicher mit einer Funktion behoben werden können, aber es gab noch weitere Gründe, die für einen erneuten Wechsel des Frameworks sprachen:

1. Die “updateData”- und “addData”-Funktionen von Tabulator würden die Daten nur in dem JSON-Objekt verändern, nicht in der Quelltable in der SQL-Datenbank.
2. Das “unechte” JSON führte zu einer Verzögerung von mehreren Stunden, woraufhin der Autor sich zwischenzeitig bereits mit der nächsten Alternative befasst hat.
3. Das Design von Tabulator ist wenig ansprechend und passt nicht gut ins Bild von CKAN.

5.3 Iteration 3: Verwendung von “rohem” Recline.js mit CSV

Die Wahl eines anderen Frameworks für den dritten Versuch führte dazu, dass eine weitere Dokumentation gelesen, verstanden und ausprobiert werden musste. Lobenderweise muss erwähnt werden, dass sich in der Dokumentation von Recline.js einige Tutorials wiederfinden, die zeigen, wie das Framework für unterschiedliche Zwecke genutzt werden kann. Dabei fiel allerdings - ähnlich wie bei CKAN, welches bekanntermaßen von derselben Organisation (OKFN) vorangetrieben wird - auf, dass einige dieser Tutorials fehlerhaft sind und wahrscheinlich aus älteren Versionen des Frameworks stammen.

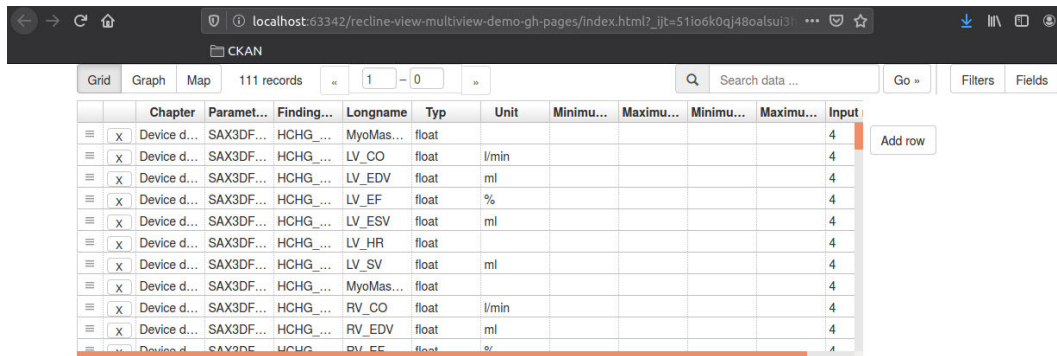
Glücklicherweise konnte funktionierender Demo-Code in Github-Repositories einer an der Entwicklung von Recline.js beteiligten Person gefunden werden [24], der eine gute Basis zur Orientierung bot.

Das gefundene Beispiel wurde auf die eigenen Bedürfnisse angepasst:

- überflüssige Darstellungsvarianten wie eine Karte für geospatiale Daten und eine Darstellung der Daten in einem Graph wurden entfernt,
- die verwendete Datenquelle wurde von einem hart programmierten Javascript-Beispiel-Array auf die gewünschte Beispiel-CSV-Datei geändert,

- der Code wurde an einigen Stellen um für den Anwendungsfall irrelevante Funktionen gekürzt.

Nachdem alle Änderungen durchgeführt wurden, war erfreulicherweise endlich ein sichtbares Ergebnis in Form einer durchsuchbaren und filterbaren Tabelle, die ihre Daten aus einer CSV-Datei bezieht, allerdings noch nicht in CKAN integriert war, zu sehen.



	Chapter	Paramet...	Finding...	Longname	Typ	Unit	Minimu...	Maximu...	Minimu...	Maximu...	Input
<input type="checkbox"/>	Device d...	SAX3DF...	HCHG...	MyoMas...	float						4
<input type="checkbox"/>	Device d...	SAX3DF...	HCHG...	LV_CO	float	l/min					4
<input type="checkbox"/>	Device d...	SAX3DF...	HCHG...	LV_EDV	float	ml					4
<input type="checkbox"/>	Device d...	SAX3DF...	HCHG...	LV_EF	float	%					4
<input type="checkbox"/>	Device d...	SAX3DF...	HCHG...	LV_ESV	float	ml					4
<input type="checkbox"/>	Device d...	SAX3DF...	HCHG...	LV_HR	float						4
<input type="checkbox"/>	Device d...	SAX3DF...	HCHG...	LV_SV	float	ml					4
<input type="checkbox"/>	Device d...	SAX3DF...	HCHG...	MyoMas...	float						4
<input type="checkbox"/>	Device d...	SAX3DF...	HCHG...	RV_CO	float	l/min					4
<input type="checkbox"/>	Device d...	SAX3DF...	HCHG...	RV_EDV	float	ml					4
<input type="checkbox"/>	Device d...	SAX3DF...	HCHG...	RV_EF	float	g					4

Abbildung 5.1: Recline.js: Darstellung & Bearbeitung einer Tabelle im Browser

An dieser Stelle ist dann aufgefallen, dass das offizielle CSV-Backend für Recline.js die Recline.js-Backend-API nicht vollständig implementiert ist und deshalb das Speichern der im Frontend durchgeführten Änderungen an den Daten nicht funktioniert. [12] Diese Funktion zu implementieren hätte den zeitlichen Rahmen der Arbeit überschritten, weshalb auf diese Funktionalität leider verzichtet werden muss.

Nun musste der angepasste Javascript-Code mithilfe der Templating-Sprache Jinja2 und dem in CKAN verwendeten Webressourcen-Publikationsframework Fanstatic in die CKAN-Oberfläche eingebunden werden. Auch an dieser Stelle gab es einige Hindernisse, weil die Dokumentation von CKAN zum Teil ungenau oder veraltet ist. Beispielsweise fügt das zu verwendende benutzerdefinierte Jinja2-Tag zum Einbinden einer mittels Fanstatic geladenen Ressource alle geladenen CSS-Dateien standardmäßig in scheinbar beliebiger Reihenfolge im “head”-Tag der im Endeffekt entstehenden HTML-Datei ein und alle Javascript-Dateien in ähnlich beliebiger Reihenfolge am Ende der HTML-Datei. [1] Für das Skript ist die richtige Reihenfolge zumindest für die Javascript-Dateien aber fundamental.

5 Implementation einer Extension

Mehr oder minder zufällig - da sich die entsprechende Seite an einer eher unerwarteten Stelle in der CKAN-Dokumentation befand, nämlich nicht in demselben Kapitel wie die sonstigen Anleitungen rund um Ressourcen wie CSS und Javascript - konnte mittels einer Config-Datei [25] eingestellt werden, in welcher Reihenfolge die Ressourcen geladen werden und somit die Extension erfolgreich in CKAN integriert werden.

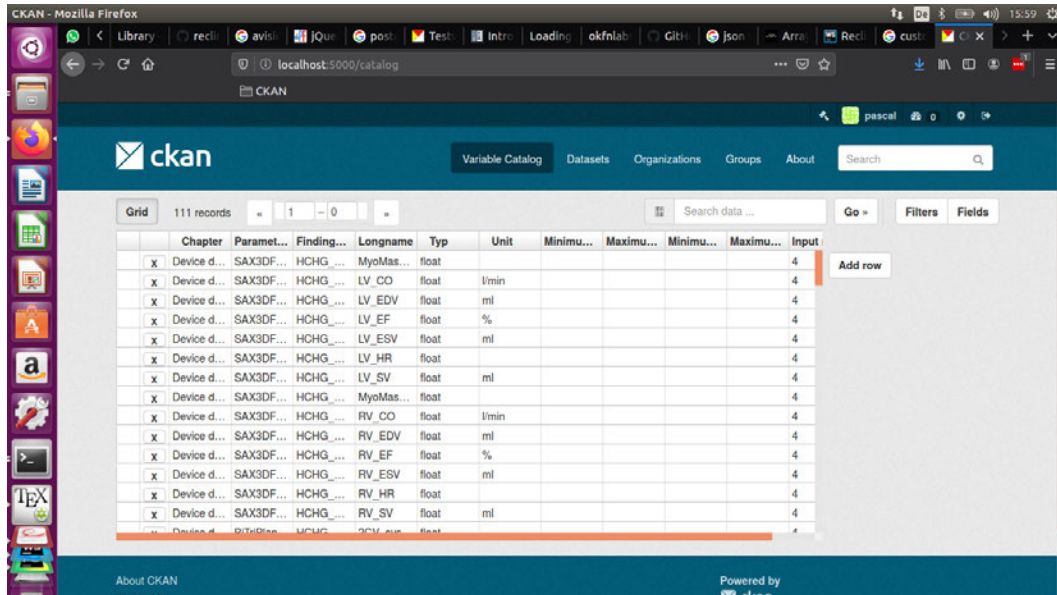


Abbildung 5.2: CKAN: erfolgreich implementierte Extension

6 Bewertung

Entgegen der in der ISO/IEC 25040 empfohlenen und in 2.4 bereits beschriebenen Evaluation in fünf Schritten, reduziert diese Arbeit die Evaluation auf vier Schritte. Der dritte Schritt, der sich mit der zeitlichen und finanziellen Planung sowie Personal- und Ressourcenplanung beschäftigt, wird in Anbetracht dessen, dass dies eine Bachelorthesis ist, an der eine Person arbeitet, außer Acht gelassen.

1	Anforderungen an die Evaluation
2	Definition eines Qualitätsmodells
3	Durchführung der Evaluation
4	Abschluss der Evaluation

Tabelle 6.1: Die vier Schritte dieser Evaluation

Diese Arbeit zielt darauf ab, zwei der führenden Produkte auf dem Gebiet der Datenportal-Plattformen für zwei Anwendungsfälle zu testen und dabei herauszufinden, welches der beiden Produkte in welchem Fall besser geeignet ist. Dabei dient der Autor dieser Arbeit sowohl als Administrator und Benutzer der Produkte, als auch als Entwickler und Evaluator. Gegenstand dieser Evaluation sind neben den Plattformen in Form von Software ebenso die jeweiligen Dokumentationen.

6.1 Definition eines Qualitätsmodells

Bei der Definition des Qualitätsmodells orientiert sich diese Arbeit am ISO/IEC-Standard 25010, wie bereits in 2.4 erwähnt. Die acht dort aufgeführten Kriterienkategorien enthalten insgesamt 31 Qualitätsmerkmale, wie in Abb. 2.6 zu sehen ist. Während des Findungsprozesses geeigneter Metriken und der tiefgehenden Beschäftigung mit den beiden Systemen hat sich für einige der dort genannten Kriterien herausgestellt, dass sie für die Bewertung im Rahmen dieser Arbeit aus verschiedenen Gründen nicht gut geeignet

sind. Diese Kriterien werden nachfolgend mit einer kurzen Begründung aufgeführt und dann in dieser Arbeit nicht weiter berücksichtigt:

1. **Functional Correctness:** Dieses Kriterium ist eher für die Bewertung einer Auftragsarbeit geeignet, bei der die überprüften Funktionen mit den Anforderungen, die in einem Lastenheft definiert wurden, abgeglichen werden können.
2. **Functional Appropriateness:** Dieses Kriterium ist eher für die Bewertung einer Auftragsarbeit geeignet, bei der die überprüften Funktionen mit den Anforderungen, die in einem Lastenheft definiert wurden, abgeglichen werden können.
3. **Fault Tolerance:** Bei Ausfall einer Software- oder Hardwarekomponente des Systems wird nicht erwartet, dass mit dem System weitergearbeitet werden kann. Daher erfolgt auch keine Bewertung in dieser Hinsicht.
4. **Testability:** Die Relevanz dieses Kriteriums fällt bei der Art der möglichen Anpassungen eher niedrig aus. Daher wird es aus Gründen der Zeitersparnis weggelassen.

Jedes der übrigen 27 Qualitätsmerkmale wird angelehnt an die Likert-Skala [19] wie folgt bewertet:

Punkte	Beschreibung	Ergebnis der Metrik
0	Das Kriterium wird nicht erfüllt.	$x = 0$
1	Das Kriterium wird eher nicht erfüllt.	$0 < x < 0,5$
2	Das Kriterium wird eher erfüllt.	$0,5 \leq x < 1$
3	Das Kriterium wird erfüllt.	$x = 1$

Tabelle 6.2: Punktesystem für die Bewertung

Für beide Produkte wird zu jedem der objektiv messbaren Merkmale eine Messung durchgeführt und dann in einem kurzen Statement festgehalten, welche Qualitätsstufe (= Punktzahl) erreicht wurde. Hierbei orientiert sich diese Arbeit stark an den von Professor Giuseppe Santucci von der Universität La Sapienza (Rom) in seinen Vorlesungsfolien [34] vorgeschlagenen Metriken, während für die subjektiveren Merkmale ausschließlich eine Einschätzung in Form eines Statements erfolgt. Die Punkte der einzelnen Qualitätsmerkmale werden dann für die Kriterienkategorien aufsummiert in einer Tabelle dargestellt, sodass eine Vergleichbarkeit der beiden Systeme gewährleistet ist. Abschließend werden

die Ergebnisse noch für die beiden Anwendungsfälle gewichtet, da sich die Anforderungen zum Teil etwas unterscheiden.

In der folgenden Tabelle werden Metriken für die einzelnen Kriterien definiert, sofern möglich. Kriterien, für die keine Metrik definiert werden konnte, werden hier nicht erneut aufgeführt.

Kriterium	Metrik
1. Functional Suitability	
1.1 <i>Functional Completeness</i>	Anzahl der erfüllbaren Aufgaben und Ziele aus 3.1.3 / Anzahl aller Aufgaben und Ziele aus 3.1.3 [34]
2. Performance Efficiency	
2.1 <i>Time Behaviour</i>	Antwortzeit des Systems für typische Aufgaben unter 1/3/5 Sekunden? ²
2.2 <i>Resource Utilization</i>	durchschnittliche CPU- & RAM-Auslastung ³
2.3 <i>Capacity</i>	Sind die maximale Dateigröße und -anzahl ausreichend? ⁴
3. Compatibility	
3.1 <i>Co-existence</i>	Läuft das System problemlos neben anderen Systemen? ⁴
3.2 <i>Interoperability</i>	Anzahl der durch die API benutzbaren Arten von Daten / Anzahl aller Arten von Daten
4. Usability	
4.1 <i>Appropriateness Recognizability</i>	Zeit, die zum Verstehen des Nutzens von Funktionalitäten benötigt wird ^{5, 6}
4.2 <i>Learnability</i>	Zeit, die zum Erlernen des richtigen Einsatzes von Funktionalitäten benötigt wird ^{5, 6}
4.3 <i>Operability</i>	Anzahl von Funktionalitäten mit max. 3 Schritten / Anzahl aller Funktionalitäten [34]
4.4 <i>User Error Protection</i>	Anzahl für den Endnutzer sichtbarer Fehler / Stunde
4.5 <i>User Interface Aesthetics</i>	nicht messbar, subjektiv ⁶
4.6 <i>Accessibility</i>	nicht messbar, subjektiv ⁶
5. Reliability	
5.1 <i>Maturity</i>	durchschnittliche Zeit zwischen Fehlern [34]
5.4 <i>Recoverability</i>	Sind verlorene Daten im Fehlerfall wiederherstellbar? ⁴
7. Maintainability	
7.4 <i>Modifiability</i>	Anzahl auftretender Fehler bei versuchter Modifizierung ⁷
8. Portability	
8.2 <i>Installability</i>	Anzahl der Probleme / Installation bei genauer Einhaltung der Installationsanleitung

1 Diese Metriken erscheinen zwar zuerst sinnvoll, aber sind eher für die Bewertung einer Auftragsarbeit geeignet als für die Bewertung fertiger Open-Source-Systeme.

2 Bewertung: Zeit $t \leq 1s$: $x = 1$; $1s < t \leq 3s$: $x = 0.67$; $3s < t \leq 5s$: $x = 0.33$; $t > 5s$: $x = 0$

3 Bewertung: \emptyset Auslastung $A < 80\%$: $x = 1$; $80\% \leq A < 90\%$: $x = 0.5$; $90\% \leq A \leq 100\%$: $x = 0$

4 Nein: $x = 0$; Ja: $x = 1$

5 subjektive Einschätzung, ob "zu lange" oder "okay".

6 Wäre mithilfe eines Usability-Tests mit Fragebögen "messbar", würde aber den Rahmen der Arbeit sprengen.

7 Die gleiche Art von Modifizierung muss gewährleistet sein.

Tabelle 6.3: Definition von Metriken

Nachfolgend ist beschrieben, wie die Kriterien ohne Metriken, also jene mit der Fußnote "6" versehenen Kriterien bewertet werden.

4.5 User Interface Aesthetics: Ob eine Interaktion als “pleasing & satisfying” wahrgenommen wird, hängt vom Empfinden einer Person ab. Der Autor empfindet die drei nachfolgenden Eigenschaften hierfür als fundamental:

1. *Der Text muss gut lesbar sein;* die Wahl angenehmer Farben sowie eines hohen Kontrasts für Schrift und Hintergrund sind hierfür wichtig, ebenso muss die Schriftgröße sich in einem angemessenen Rahmen befinden.
2. *Schaltflächen müssen eindeutige Funktionen erfüllen;* die Beschriftung der Schaltflächen muss möglichst eindeutig sein, sodass klar ist, welche Konsequenz aus einer Aktion folgt und die Software intuitiv bedienbar ist.
3. *Der Benutzer darf beim Umgang mit der Software keinen Stress empfinden;* es ist für eine positive Erfahrung mit einem System nicht förderlich, wenn der Benutzer unablässig das Gefühl hat, Schaden anrichten zu können.

4.6 Accessibility: Da im Rahmen dieser Arbeit kein Usability-Test durchgeführt werden kann, ist es nicht möglich zu überprüfen, ob Personen mit verschiedenen guten Fähigkeiten im Umgang mit Softwaresystemen mit den für sie typischen Aufgaben zurechtkommen. Der Autor wird versuchen, sich in folgende typische Personengruppen hineinzusetzen und einzuschätzen, wie gut sie ihre Aufgaben erfüllen können:

1. Studierende, die Daten für Abschlussarbeiten nutzen wollen (eher computeraffin, nur HAW);
2. Wissenschaftler, die Projekte/Daten veröffentlichen und verwenden wollen (nicht unbedingt computeraffin, HAW + HCHS);
3. Administratoren, die die Systeme verwalten wollen (typischerweise computeraffin, HAW + HCHS).

5.2 Availability: Sofern keine erkennbar softwareseitig verschuldeten Downtimes vorliegen, ist anzunehmen, dass die Verfügbarkeit bei einer Webapplikation - wozu beide untersuchten Systeme gehören - grundsätzlich eher von der darunterliegenden Architektur abhängig ist. Bewertet wird daher sporadisch - also negativ bei Auftreten von softwareseitigen Problemen, die die Verfügbarkeit des Services beeinträchtigen und ansonsten positiv.

6.1 Confidentiality: Die Beantwortung einer Reihe von Fragen gibt Aufschluss darüber, ob ein System Daten vor unberechtigtem Zugriff schützt:

- Gibt es eine generelle Zugriffsbeschränkung (Login)?
- Auf welche Weise kann der Login erfolgen (Benutzername/Passwort, OAuth, Shibboleth, ...)?
- Können einzelne Projekte/Datensätze gegen den Zugriff von registrierten Nutzern - welche also generell einen Zugang zum System haben - geschützt werden?

6.2 Integrity: Die Frage, ob die beiden untersuchten Systeme vor unautorisierter Veränderung des Source Codes und der Daten schützt, ist - da beide Systeme Webanwendungen sind - abhängig von der darunterliegenden Architektur. Da in beiden Systemen zum Teil unterschiedliche Subsysteme (Webserver, Datenbanksysteme, Search Engines) verwendet werden, werden für die Bewertung dieses Kriteriums die Subsysteme mittels Recherche auf bekannte Sicherheitslücken geprüft und anhand dessen bewertet.

6.3 Non-repudiation & 6.5 Accountability: Um diese Kriterien zu erfüllen, müssen die Systeme festhalten, welcher User zu welcher Zeit welche Events oder Aktionen auslöst bzw. durchführt, z.B. mittels einer oder mehrerer Logdateien oder entsprechenden Log-Einträgen in den Datenbanken. In die Bewertung wird neben der grundsätzlichen Frage ob ein solches Logging existiert auch die Genauigkeit des Logs einfließen.

6.4 Authenticity: Der Grad der Authentizität eines Benutzers hängt von der Art und Weise des Logins ab. Normale Benutzername-Passwort-Kombinationen sind grundsätzlich schwächer als z.B. eine Zweifaktor-Authentifizierung. Natürlich ist auch die Erfüllung dieses Kriteriums, da die Systeme online sind, abhängig von der generellen Sicherheit, die die darunterliegende Architektur bietet.

Die Authentizität der den Benutzern zur Verfügung gestellten Ressourcen hängt davon ab, wie die Logins vergeben werden. Kann jeder sich auf der Startseite selbst einen Account erstellen und Projekte/Datensätze hochladen oder werden die Accounts von einer vertrauenswürdigen Stelle erstellt/verifiziert?

7.1 Modularity: Um zu überprüfen, inwiefern die Systemkomponenten voneinander abgekapselt sind und nahtlos durch äquivalente Komponenten ersetzt werden können, wird der Source Code stichprobenartig überprüft:

- Wurden Interfaces definiert und ordnungsgemäß verwendet?
- Wird durch die Dokumentation und/oder Kommentare klar, welche Komponenten voneinander abhängen?

7.2 Reusability: Wenn Komponenten der Anwendungen leicht wiederverwendet werden können, ist es schneller und einfacher möglich, die Systeme für die eigenen Bedürfnisse anzupassen. Auch hierfür sind wohldefinierte und ordnungsgemäß benutzte Interfaces, aber auch eine gut ausgestaffierte API gute Anzeichen.

8.1 Adaptability & 8.2 Installability: Wie gut die Softwaresysteme in verschiedenen Umgebungen bestehend aus unterschiedlicher Hard- und Software installiert werden können und dann auch im simulierten Betrieb funktionieren, wird evaluiert indem die Systeme auf Rechnern mit verschiedenen Spezifikationen installiert und ausgeführt werden. Auch hierzu bieten sich einige Fragen an:

- Auf welchen Betriebssystemen kann das System (nicht) ausgeführt werden?
- Was ist die Untergrenze der Leistungsfähigkeit, die die Hardware benötigt, um das System zufriedenstellend ausführen zu können?
- Funktionieren bestimmte Hardware- oder Software-/Betriebssystem-Spezifikationen evtl. gar nicht?

8.3 Replaceability: Ist eines der Systeme in der Lage, das andere System zu ersetzen?

6.2 Durchführung der Evaluation

In diesem Unterabschnitt wird beschrieben, wie die beiden Systeme CKAN und Dataverse auf die in 2.4 vorgestellten Kriterien überprüft und nach den in 6.1 festgelegten Metriken und Fragestellungen bewertet wurden.

6.2.1 CKAN

In diesem Abschnitt erfolgt die Bewertung des Systems CKAN. Dank des klar definierten Templates in Form der ISO/IEC 25010 und des in 6.1 ausführlich beschriebenen Qualitätsmodells verlief die Bewertung des Systems reibungslos.

6.2.1.1 Functional Suitability (Funktionelle Eignung)

Functional Completeness: Um die funktionelle Vollständigkeit zu überprüfen, wird nachfolgend für jedes Ziel und jede Aufgabe, das bzw. die in 3.1.3 definiert wurden, beschrieben, inwiefern es/sie von CKAN erfüllt wird.

[Z01] Abbildung der Organisationshierarchie im Berechtigungssystem:

In CKAN gibt es sogenannte “Sysadmins”, also Systemadministratoren und reguläre Benutzer. Sysadmins können im Gegensatz zu regulären Benutzern die in 2.2.2 bereits erklärten Organizations erzeugen. Ein regulärer Benutzer kann, sofern er noch keiner Organization angehört, nichts tun, außer sich auf “public” gesetzte Datasets anzuschauen und herunterzuladen. Um einer Organization beizutreten muss ein regulärer Benutzer von einem Admin der Organization (\neq Sysadmin) hinzugefügt werden. Je nach zugewiesener Rolle kann der Benutzer nun weitere Operationen ausführen (vgl. 2.2.2).

Die Organisationsstruktur der HAW könnte zum Beispiel auf diese Weise in CKAN simpel nachmodelliert werden:

CKAN-Rolle	Rolle in der HAW
Sysadmin	ITSC-Mitarbeiter
Organization Admin (z.B. Departments, Forschungsgruppen)	Professoren, ggf. Wissenschaftl. Mitarbeiter
Organization Editor	Wissenschaftl. Mitarbeiter, ggf. Studenten
Organization Member	z.B. externe Zweitprüfer, die Zugriff auf nicht veröffentlichte (z.B. NDA-pflichtige) Datasets und damit verbundene Arbeiten benötigen, ohne diese verändern zu dürfen

Für die HCHS könnte das Berechtigungssystem wie folgt genutzt werden:

CKAN-Rolle	Rolle im HCHS-Anwendungsfall
Sysadmin	IT-Mitarbeiter der Studie
Organization Admin (z.B. Forschungsgruppen)	Leiter bzw. Ansprechpartner der Forschungsgruppe
Organization Editor	weitere Wissenschaftler der Forschungsgruppe
Organization Member	z.B. die Studienzentrumsleitung, um (noch) nicht veröffentlichte Datasets und damit verbundene Arbeiten einsehen zu können

Das Ziel wird als erfüllt angesehen.

[Z02] Zugriff auf bestimmte Daten/Dateien/Projekte nur durch bestimmte Nutzer:

Jeder Besucher einer CKAN-Seite kann jedes **veröffentlichte** Dataset einsehen, unabhängig davon, ob er registriert ist oder nicht. Jedes Dataset kann jedoch von Mitgliedern der es besitzenden Organization auch auf “private” gestellt werden. In diesem Fall ist das Dataset nur für registrierte Nutzer, die ebenfalls Mitglieder der Organization sind sichtbar, allerdings unabhängig der Rolle, die das Mitglied in der Organization innehat.

Das Ziel wird als erfüllt angesehen.

[Z03] geordnete Sammlung aller erhobenen Forschungsdaten:

CKAN unterbindet standardmäßig das Hinzufügen von Datasets außerhalb von Organizations. Hierdurch wird eine gewisse Grundordnung gewahrt, da alle hochgeladenen Datasets einer Organization zugewiesen sind. Wenn zudem noch einige Nutzungsregeln (z.B. “jedes Dataset muss mindestens drei Tags erhalten” oder “jedes Dataset muss eine mindestens 100 Zeichen lange Beschreibung erhalten”) bestimmt werden, liefert CKAN eine gute Grundlage zum ordentlichen Sammeln von Forschungsdaten. Das Ziel wird als erfüllt angesehen.

[Z04] Kategorisierung von Projekten (Tagging):

CKAN ermöglicht das Kategorisieren von Datasets mittels Tags, nach denen gesucht und gefiltert werden kann. Das Ziel wird als erfüllt angesehen.

[Z05] Zuordnung von schriftlichen Ergebnissen zu Daten und Dateien:

Ein Dataset in CKAN kann mehrere Dateien enthalten, also kann dem Dataset immer auch ein Textdokument mit schriftlichen Ergebnissen beigefügt werden. Das Ziel wird als erfüllt angesehen.

[Z06] Teilen von Forschungsdaten mit anderen Forschern:

Jemand, der seine Datasets mit anderen Forschern/Studenten auf der Plattform teilen möchte, kann sie entweder auf “öffentlich” setzen, und somit *unkontrolliert* für die gesamte Plattform freigeben, oder gezielt Benutzer mit der Rolle “Member” in seine Organization einladen, um die Datasets nur mit bestimmten Benutzern zu teilen. Das Ziel wird als erfüllt angesehen.

[Z07] Abbildung eines Variablenkatalogs unabhängig von tatsächlich hochgeladenen Dateien (nur HCHS):

Diese Anforderung wird von CKAN nicht erfüllt. Man könnte eine etwas unschöne Lösung ohne Programmieraufwand finden, indem eine Tabellendatei mit Metadaten zu den

eigentlichen Dateien mit in das Dataset hochgeladen und z.B. “Variable Information Table” genannt wird, aber die Idee hinter der Anforderung war ein zentrales Register aller Variablen, die in der Studie erhoben und für die Wissenschaftler, die die Plattform nutzen, verfügbar gemacht werden können. Die geforderte Übersichtlichkeit bliebe so auf der Strecke. Das Ziel wird als nicht erfüllt angesehen.

[A01] Upload vieler Dateien in einem Projekt:

In CKAN kann ein Dataset zwar unbegrenzt viele zugehörige Dateien haben, aber das Webinterface bietet in der Standardinstallation keinen Upload mehrerer Dateien gleichzeitig an. Das Problem kann zwar zum Beispiel durch das Zusammenfügen der Dateien zu einem ZIP-Archiv o.Ä. umgangen werden, aber dann können die Dateien z.B. nicht online in der Preview angesehen werden, was eigentlich ein zentraler und schöner Vorteil von CKAN ist. Es gibt jedoch bereits eine Extension namens “ckanext-file-uploader-ui”, die das UI modifiziert, sodass mehrere Dateien gleichzeitig hochgeladen werden können. Alternativ lässt sich mithilfe der API auch unkompliziert eine eigene - wahrscheinlich rudimentärere - Lösung schaffen. Die Aufgabe wird als erfüllbar angesehen.

[A02] Upload von Dateien bis 10GB Größe:

Die maximale Dateigröße kann in einer Konfigurationsdatei festgelegt werden. Wird keine Maximalgröße festgelegt, gibt es auch keine Grenze. Die Aufgabe wird als erfüllbar angesehen.

[A03] Suchen und Finden von Dateien:

Die Suchleiste, die sich gut sichtbar in der Menüleiste im Header der Benutzeroberfläche wiederfindet, benutzt im Hintergrund als Search-Engine mit Apache Solr eines der zwei am häufigsten benutzten Produkte auf dem Gebiet [2]. Ein kurzer Test bestätigt, dass die Suche nach einem Dateinamen sogar bei Unvollständigkeit zu einem positiven Ergebnis führt. Es kann also nicht nur nach Datasets, sondern auch nach einzelnen Dateien, die sich in den Datasets befinden, gesucht werden. Die Aufgabe wird als erfüllbar angesehen.

[A04] Suchen und Finden von Projekten:

Projekte sind in CKAN *Datasets*. Diese lassen sich ebenso gut wie einzelne Dateien über die Eingabe des nicht zwingendermaßen vollständigen Namens in der Suchleiste finden und es werden auch Datasets angezeigt, die ein übereinstimmendes Tag mit dem Suchbegriff haben. Die Aufgabe wird als erfüllbar angesehen.

[A05] Suchen und Finden von Benutzergruppierungen:

Benutzergruppierungen heißen in CKAN *Organizations*. Über die Suchleiste werden nur Organizations angezeigt, die bereits mindestens ein Dataset veröffentlicht haben. Die Schaltfläche “Organizations” in der Menüleiste führt zu einer Übersichtsseite, auf der alle Organizations angezeigt werden. Hier gibt es eine eigene Suchleiste, über diese können alle Organizations von allen registrierten Benutzern gefunden werden. Die Aufgabe wird als erfüllbar angesehen.

[A06] Suchen und Finden von einzelnen Benutzern:

Die Suche nach einzelnen Benutzern ist nicht besonders intuitiv gestaltet. Über die Suchleiste in der Menüleiste werden Benutzer nicht angezeigt. Zudem gibt es keine Schaltfläche in der Menüleiste, über die andere Benutzer angezeigt oder gesucht werden können. Die einzige Möglichkeit, über das UI zur Benutzerübersicht zu kommen, ist indem man auf die Schaltfläche mit dem eigenen Namen (Tooltip: View Profile) klickt und dann auf “Users”. Hier gibt es dann wiederum auch eine Suchleiste, in der man Benutzer nach ihrem Namen suchen kann. Die API liefert im Gegensatz zur Weboberfläche eine simple Funktion zum Abruf einer Benutzerliste. Die Aufgabe wird trotz der Umständlichkeit als erfüllbar angesehen.

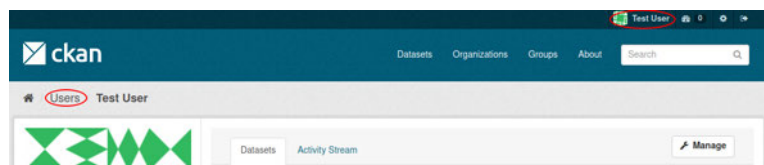


Abbildung 6.1: CKAN: Nicht sehr intuitive Lokation der Benutzerübersicht

[A07] Download von Dateien mit entsprechender Autorisierung:

Alle Dateien, die Teil eines veröffentlichten Datasets sind, sind in einer Standard-CKAN-Instanz für jeden Besucher der Seite einseh- und herunterladbar. Hierfür muss der Besucher sich noch nicht einmal registrieren. Es gibt zwar eine Reihe von Optionen für die

Autorisierung, die in einer Konfigurationsdatei vorgenommen werden können, aber ohne eine Veränderung am Source Code vorzunehmen ist es nicht möglich, dies zu unterbinden. Es ist jedoch möglich, Datasets nicht zu veröffentlichen. Dann sind die Datasets nur für Mitglieder der es besitzenden Organization einseh- und herunterladbar. Die Aufgabe wird als erfüllbar angesehen.

[A08] Verwalten von einzelnen Benutzern:

Das Verwalten einzelner Benutzer ist für Sysadmins in der Benutzeroberfläche möglich. Sobald man herausgefunden hat, wie die Benutzerübersicht zu erreichen ist, ist das Verwalten einzelner Benutzer sehr einfach. Über eine “Manage”-Schaltfläche lassen sich alle Informationen zu dem Benutzer mit Ausnahme des Benutzernamens verändern. Zudem kann hier das Passwort des Benutzers zurückgesetzt werden, sein API-Key neu generiert werden oder der Account gelöscht werden. Die Aufgabe wird als erfüllbar angesehen.

[A09] Verwalten von Benutzern in Gruppen:

Das Verwalten mehrerer Benutzer gleichzeitig in der Weboberfläche ist in CKAN ohne eine Extension oder Veränderung des Source Codes nicht möglich. Die API bietet jedoch ein umfangreiches Paket an Funktionen an, mit denen das Erstellen, Verändern und Löschen von mehreren Benutzeraccounts gleichzeitig möglich gemacht werden kann. Die Aufgabe wird als nicht erfüllbar angesehen, da die Funktionalität nicht ohne Programmieraufwand oder direkte Arbeit in der Konsole auskommt.

[A10] Veröffentlichen von Forschungsergebnissen:

Sobald ein Dataset auf “public” gesetzt wird, ist es von jedem Besucher der Hauptseite in der Suche auffindbar. Also kann jeder Benutzer, der seine Ergebnisse veröffentlichen möchte, einfach seine zugehörigen Datasets in seiner Arbeit erwähnen und interessierte Leser mithilfe eines Links zu den zugehörigen Dateien führen. Die Aufgabe wird als erfüllbar angesehen.

[A11] Bearbeiten des Online-Variablenkatalogs im Browser (nur HCHS):

Da in der Standard-CKAN-Installation die Abbildung eines Online-Variablenkatalogs nicht möglich ist (s. Z07), ist das Bearbeiten des Katalogs im Browser folglich auch nicht

möglich.

Die Erkenntnisse noch einmal kurz in einer Tabelle zusammengefasst:

Ziel/Aufgabe	wird von CKAN erfüllt
Z01	Ja
Z02	Ja
Z03	Ja
Z04	Ja
Z05	Ja
Z06	Ja
Z07	Nein
A01	Ja
A02	Ja
A03	Ja
A04	Ja
A05	Ja
A06	Ja ¹
A07	Ja
A08	Ja
A09	Nein ²
A10	Ja
A11	Nein

1: aber umständlich

2: mit API ja

Tabelle 6.4: CKAN: Functional Completeness

Aus dem Ergebnis 15x ja und 3x nein ($\frac{15}{18} \approx 0.83$) ergibt sich nach Tabelle 6.2 eine Bewertung von $\mathcal{2} = \text{eher erfüllt}$.

6.2.1.2 Performance Efficiency (Performanz)

Time Behaviour: Alle Ladezeiten auf der Plattform bleiben regelmäßig unter einer Sekunde, mit Ausnahme vom Dateiupload bei größeren Dateien, was aber abhängig von der Upload-Rate der genutzten Internetverbindung ist. Falls ein Benutzer beim Upload größerer Dateien nicht längere Zeit auf das Aktualisieren der Weboberfläche warten möchte, kann er den Upload auch unter Benutzung der API-Funktion “resource_create” durchführen. Das Kriterium wird mit $\mathcal{3} = \text{erfüllt}$ bewertet.

Resource Utilization: In der nachfolgenden Tabelle wird aufgeführt, wie hoch die CPU- und RAM-Auslastungen während typischer Aufgaben auf dem Testsystem (Ubuntu 18.04, 4GB RAM, 2 CPU-Kerne) liegen. Hierfür wurden mittels des Programms “htop” Durchschnittswerte über den Zeitraum von einer Minute genommen. Hierbei half das Attribut “Load average”, welches die durchschnittliche Anzahl aktiver Prozesse der letzten Minute anzeigt. Da der virtuelle Server über zwei CPU-Kerne verfügt, muss der Load Average halbiert werden um daraus die CPU-Auslastung zu lesen, da bei voller Auslastung beider Kerne der Load Average 2.0 betragen würde (durchgängig zwei aktive Prozesse = 100% CPU-Auslastung bei zwei Kernen).

Aufgabe	CPU	RAM
Leerlauf ohne CKAN	3%	7%
Leerlauf inkl. CKAN im Leerlauf	5.0%	15.3%
Upload einer großen Datei	6.5%	17.5%
mehrere gleichzeitige Suchabfragen ¹	20.5%	19.0%
mehrere gleichzeitige Downloads ¹	6.0%	19.1%

1: Diese Aufgabe wurde mithilfe der API simuliert

Tabelle 6.5: CKAN: Resource Utilization

Bei allen Belastungstests blieben die Auslastungen der CPU und des RAM weit unter 80%, daraus ergibt sich eine Bewertung von $\mathfrak{S} = \text{erfüllt}$.

Capacity: Die maximale Anzahl von Dateien sowie die maximale Dateigröße sind unlimitiert. Daraus ergibt sich eine Bewertung von $\mathfrak{S} = \text{erfüllt}$.

6.2.1.3 Compatibility (Kompatibilität)

Co-existence: Um zu überprüfen, ob CKAN neben einem oder mehreren anderen Systemen laufen kann, ohne Schwierigkeiten zu bereiten, wurde ein LAMP¹-Stack installiert und auf Port 80 mit einer kleinen Beispiel-Website zum Laufen gebracht. Solange der Verwender die Möglichkeit hat, über die verfügbaren (also offenen) Ports des Servers zu verfügen, laufen die Systeme problemlos nebeneinander. Daraus ergibt sich eine Bewertung von $\mathfrak{S} = \text{erfüllt}$.

Interoperability: Die CKAN-API enthält API-Calls für alle vorstellbaren Arten von Daten zu Datasets, Organizations, Groups, Benutzern und unzähligen weiteren Objekten

¹Linux, Apache, MySQL, PHP

und Eigenschaften. Die nachfolgende Tabelle zeigt einen Auswahl der Arten von Daten, die mithilfe von API-Calls abgerufen werden können.

Datenart	API-Call
(Meta-)Daten zu bestimmten - Datasets - Dateien - Groups - Organizations - Benutzern	package_show resource_show group_show organization_show user_show
Listen aller - Datasets - Groups - Organizations - Benutzer - verwendeter Tags - verwendeter Lizenzen	package_list group_list organization_list user_list tag_list license_list
Activity Streams* von - Benutzern - Datasets - Groups - Organizations	user_activity_list package_activity_list group_activity_list organization_activity_list

* Auflistung aller Aktivitäten des betrachteten Objekts

Tabelle 6.6: CKAN: Interoperability

Zudem können nicht nur bestehende Daten mithilfe der API abgerufen werden, sondern auch Datasets, Organizations, Groups und Benutzer mithilfe der API hinzugefügt werden. Im Grunde kann sich jeder Betreiber einer CKAN-Instanz also sein eigenes “Kontrollzentrum” mithilfe der API programmieren, das dann auch den Bedürfnissen entsprechend unterschiedlich mächtig sein kann.

Aus alledem ergibt sich eine Bewertung von $\mathcal{B} = \text{erfüllt}$.

6.2.1.4 Usability (Benutzbarkeit)

Appropriateness Recognizability: Der Autor konnte alle wichtigen Systemfunktionen auf Anhieb verstehen, es gibt nur zwei Kritikpunkte:

1. Die Bezeichnung “Group” für eine Sammlung von Datasets ist irreführend, aber auch dies konnte nach einmaligem Ausprobieren klargestellt werden und hätte auch vorab durch aufmerksameres Lesen der Dokumentation bereits klar sein können.
2. Der Fakt, dass die Benutzerübersicht nicht direkt aufrufbar ist, sondern nur versteckt über das eigene Profil, erschwert anfänglich die Erfüllung von Aufgaben, die in den Bereich “Benutzerverwaltung” fallen.

Obgleich beide Kritikpunkte eher unbedeutend erscheinen, sind es dennoch Kritikpunkte und eine Bewertung mit voller Punktzahl ist nicht gerechtfertigt. Daher ergibt sich eine Bewertung von $2 = \textit{eher erfüllt}$.

Learnability: Der Autor hatte wenig Mühe, die richtige Anwendung von Systemfunktionen zu erlernen. Die meisten Funktionalitäten sind so eindeutig benannt, dass bereits vor der ersten Benutzung klar ist, was das zu erwartende Resultat ist. Das Einarbeiten eines neuen Benutzers durch Ausprobieren aller Funktionen kann prinzipiell sogar im Produktivsystem geschehen, da zum Beispiel ein in der Webanwendung gelöscht Dataset nicht vollständig gelöscht, sondern nur als gelöscht markiert (und dadurch nicht mehr angezeigt) wird. Daher ergibt sich eine Bewertung von $3 = \textit{erfüllt}$.

Operability: In der nachfolgenden Tabelle steht für jede der wahrscheinlich am häufigsten verwendeten CKAN-Funktionen das Ergebnis der Überprüfung, ob sie in drei Schritten erfüllbar ist. Dabei wird jeweils davon ausgegangen, dass der Benutzer bereits zu einer für die Aufgabe angemessenen Seite navigiert ist (z.B. beim “Hinzufügen eines Benutzers zu einer Organization” auf der Hauptseite der gewünschten Organization).

Funktionalität	≤ 3 Schritte
Suchen und Finden eines Datasets	ja
Suchen und Finden einer Organization	ja
Suchen und Finden einer Group	ja
Suchen und Finden eines Benutzers	ja
Anlegen eines neuen Datasets	ja
Anlegen einer neuen Organization	ja
Anlegen einer neuen Group	ja
Anlegen eines neuen Benutzers	nein ¹
Hinzufügen einer Datei zu einem bestehenden Dataset	nein
Entfernen einer Datei aus einem Dataset	nein
Bearbeiten eines Datasets	ja
Löschen eines Datasets	ja
Hinzufügen eines Benutzers zu einer Organization	nein
Entfernen eines Benutzers aus einer Organization	ja
Zurücksetzen eines Benutzerkennworts	ja
Zurücksetzen eines API-Keys	ja
Previewen einer Datei	ja
Herunterladen einer Datei	ja
Hinzufügen eines Datasets zu einer Group	ja ²
Entfernen eines Datasets aus einer Group	ja ²

1 Das Anlegen eines neuen Benutzers ist für einen Sysadmin in der Weboberfläche nicht möglich. Per Command Line kann ein Benutzer mithilfe eines einzelnen Befehls erstellt werden, aber die Usability-Bewertung bezieht sich grundsätzlich auf die Weboberfläche. Wenn die Selbstregistrierung durch einen neuen Benutzer aktiviert ist, ist die Erstellung eines neuen Nutzers grundsätzlich in drei Schritten möglich, aber hierbei geht es um die Erstellung eines Accounts durch einen Administrator für einen Dritten.

2 Ein Dataset kann in der Weboberfläche nur über die Seite des Datasets zu einer Group hinzugefügt oder aus einer Group entfernt werden, nicht etwa über die Seite der Group mithilfe eines Suchfelds oder einer Liste. Dieser Umstand ist etwas verwirrend, aber wenn ein Benutzer über etwas Know-how verfügt, kann diese Aufgabe in drei Schritten erfüllt werden.

Tabelle 6.7: CKAN: Operability

Aus dem Ergebnis 16x ja und 4x nein ($\frac{16}{20} = 0.8$) ergibt sich nach Tabelle 6.2 eine Bewertung von $2 = \textit{eher erfüllt}$.

User Error Protection: Der Benutzer wird ausreichend vor dem Begehen von Fehlern geschützt. Dafür sorgen Pop-ups, die bei kritischen Operationen wie dem Löschen von Entitäten wie Datasets oder Organizations oder dem Zurücksetzen von API-Keys auftreten und fragen, ob die Operation wirklich ausgeführt werden soll. Zudem werden gelöschte Entitäten nicht wirklich gelöscht, sondern als gelöscht markiert. Diese Entitäten werden dann in keinen Such- und Übersichtsseiten mehr angezeigt, aber sind noch existent und können - auf etwas umständliche Art und Weise, da es keine Übersichtsseite zu gelöschten Datasets gibt - wiederhergestellt werden. Daraus ergibt sich für dieses Kriterium eine Bewertung von $\mathcal{3} = \text{erfüllt}$.

User Interface Aesthetics: Wie in 6.1 definiert, gibt es einige Unterkriterien, die für den Autor eine “pleasing & satisfying interaction” ausmachen:

1. Die Textlesbarkeit ist hervorragend. Die Farbwahl beschränkt sich in jeglicher Hinsicht auf matte, sanfte Farben; die Standard-Hintergrundfarbe ist ein sehr helles Grau, während die Standard-Schriftfarbe ein sehr dunkles Grau ist, womit auch das Kriterium eines hohen Kontrasts zwischen Hintergrund und Schrift gewährleistet ist. Auch bei Schaltflächen (standardmäßig ein dunkles Blau) hebt sich die Beschriftung (weiß) gut vom Hintergrund ab. Die Schriftgröße ist ebenfalls angemessen gewählt und durch Responsive Design ist die Webanwendung auch auf mobilen Geräten gut verwendbar.

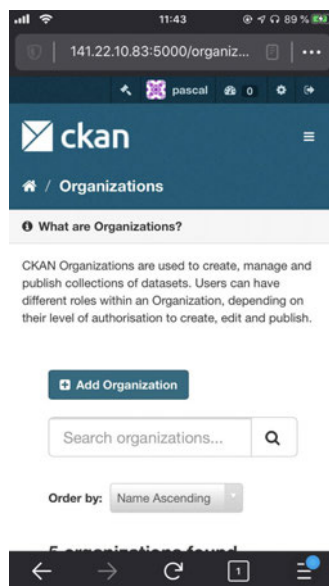


Abbildung 6.2: CKAN: Responsive Design ermöglicht Verwendung an Mobilgeräten

- Die meisten Schaltflächen sind so benannt, dass eindeutig klar ist was passiert, wenn sie betätigt werden (z.B. ‘Add Dataset’, ‘Regenerate API-Key’, ‘Update Profile’). Es gibt allerdings in der Menüleiste einige Buttons, die nur Symbole enthalten. Der Platz in der Leiste hätte zusätzlich beschreibenden Text wie ‘Settings’, ‘Dashboard’ oder ‘Logout’ für Viewports von Desktop-Format definitiv zugelassen; hiervon wurde wohl absichtlich abgesehen, denn Text-Beschreibungen finden sich als Tooltips wieder. Letztendlich ist dies eine Designentscheidung und nach einigen Minuten der Benutzung klar, aber für die Eindeutigkeit bei der ersten Benutzung wären einzelne gut gewählte Worte neben den Symbolen hilfreich.



Abbildung 6.3: CKAN: Buttons in der Menüleiste bestehen nur aus Symbolen

- Im Großen und Ganzen ist das Wohlbefinden des Benutzers durch die bereits genannten Punkte auch bei längerer Verwendung der Weboberfläche gesichert. Stress und Fehler durch gedankliche Abwesenheit werden zum Beispiel auch verhindert, indem Buttons, die kritische Aktionen nach sich ziehen, mit Signalfarben wie gelb und rot für Aktionen wie ‘Regenerate API-Key’ und ‘Delete Dataset’ eingefärbt sind. Zudem existiert bei solchen Aktionen eine doppelte Absicherung durch ein Pop-up-Fenster, das zur Bestätigung der Aktion auffordert.

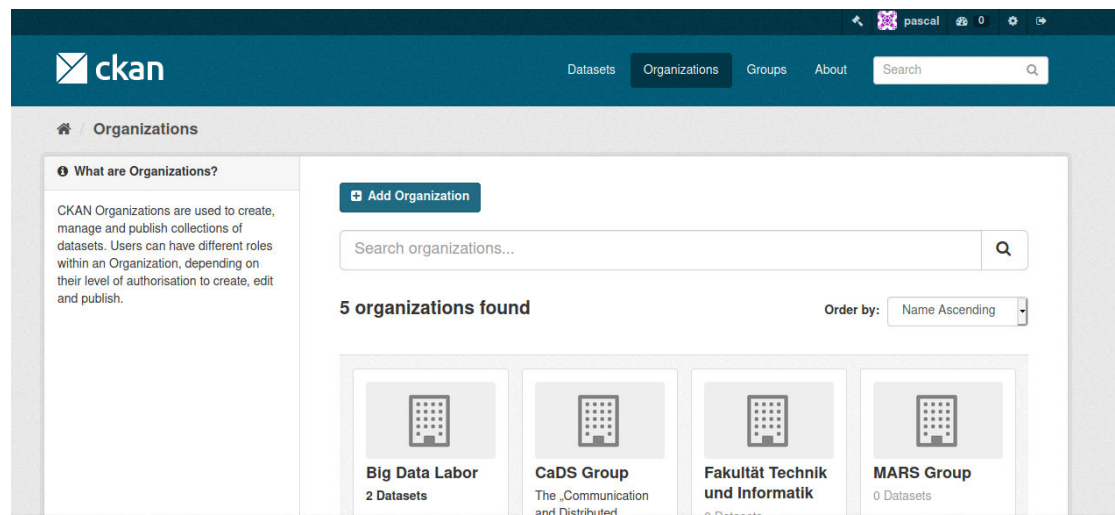


Abbildung 6.4: CKAN: angenehme Farben sorgen für Wohlbefinden bei der Verwendung

Obwohl der Autor eine andere Art von Schaltflächen für die Menüleiste präferieren würde, ist dies kein Grund, die Bewertung für dieses Kriterium herabzustufen. Abgesehen von

diesem einen Punkt gibt es absolut keine Einwände zur Benutzeroberfläche, die ansonsten ein sehr angenehmes Arbeiten ermöglicht. Hieraus ergibt sich eine Bewertung von $\mathfrak{3} = \text{erfüllt}$.

Accessibility: Alle der in 6.1 beispielhaft aufgezählten möglichen Benutzergruppen sollten alle vorstellbaren Aufgaben mit ein wenig Übung bewältigen können. Wie in den vorherigen Kriterien dieser Kategorie schon erwähnt, ist die Benutzeroberfläche eindeutig und größtenteils übersichtlich und Fehlern wird durch “Bestätigungs-Pop-ups” und Deaktivieren statt richtigem Löschen entgegengewirkt. Zudem gibt es eine Option, die Sprache benutzergebunden in eine von 50 verschiedenen Sprachen zu verändern, falls jemand mit der englischen Sprache nicht besonders vertraut ist.

Systemadministratoren einer in Docker-Containern laufenden CKAN-Instanz sollten etwas Erfahrung im Umgang mit Docker selbst und Command-Line-Interfaces haben und zumindest grob wissen, wie sie sich in einem Linux-Dateisystem zurechtfinden. Die anderen beiden erwähnten Benutzergruppen (Studierende und Wissenschaftler) müssen sich nur mit der Weboberfläche auseinandersetzen können, wozu keine besonderen Fähigkeiten vonnöten sind. Daraus ergibt sich eine Bewertung von $\mathfrak{3} = \text{erfüllt}$.

6.2.1.5 Reliability (Zuverlässigkeit)

Maturity: Nach ca. 25 Stunden aktiver Nutzung und zum Zeitpunkt des Schreibens fünf Wochen dauerhaftem Betrieb der CKAN-Instanz auf dem Webserver ist kein gravierender Fehler aufgetreten. Worauf jedoch geachtet werden muss, ist dass nach einem Docker-Update auf dem System die CKAN-Container neu gestartet werden müssen, was aber zum Beispiel mit einem Health Check in Form eines Cronjobs automatisiert werden könnte. Dies sorgt in der Gesamtheit für eine Bewertung von $\mathfrak{3} = \text{erfüllt}$.

Availability: Wie im Kriterium *Maturity* bereits erwähnt, werden die CKAN-Container bei einem Docker-Update oder anderweitigem Neustart des Docker-Services nicht automatisch neu gestartet. Davon abgesehen gab es keine Probleme mit der Verfügbarkeit des Systems, woraus eine Bewertung von $\mathfrak{3} = \text{erfüllt}$ folgt.

Recoverability: Um dieses Kriterium zu überprüfen, wurden die Docker-Container gestoppt, während ein neues Dataset in der Weboberfläche erzeugt wurde. Unabhängig davon, bei welchem Schritt der Erzeugung die Container gestoppt wurden, war nach erneutem Hochfahren der Container das Fortfahren an gleicher Stelle möglich. Der Fakt, dass gelöschte Entitäten nur als gelöscht geflaggt und nicht tatsächlich gelöscht werden

fällt hinsichtlich der Wiederherstellbarkeit auch positiv auf. Daraus folgt eine Bewertung von $\mathfrak{3} = \text{erfüllt}$.

6.2.1.6 Security (Sicherheit)

Confidentiality: Grundsätzlich kann jeder Besucher der Hauptseite, ob eingeloggt oder nicht, auf Datasets zugreifen, die auf “public” gesetzt wurden und auch Informationen zu den angelegten Organizations und Groups abrufen und tatsächlich sogar die vollständige Benutzerliste einsehen, wenn er den richtigen Link ([CKAN_URL]/user) errät. Zwar lässt sich die “Registrieren”-Funktion in einer Konfigurationsdatei deaktivieren, aber damit kann nur gesteuert werden, wer neue Datasets erstellen kann, nicht wer welche abrufen kann. Um dies zu erreichen, müsste die Anwendung z.B. in einem Intranet gehostet werden oder der Source Code dahingehend verändert werden, dass ein Login erforderlich ist, bevor weitere Funktionen der Anwendung verwendbar sind. Ohne Verwendung von Extensions ist die klassische Authentifizierung per Benutzername und Passwort die einzige Login-Methode.

Einzelne Datasets können nur für bestimmte Nutzer sichtbar gemacht werden, indem das Dataset auf “private” gesetzt und die gewünschten Nutzer der Organization hinzugefügt werden, die das Dataset besitzt.

Zwar sind die in diesem Abschnitt genannten Eckpunkte aus Open-Science-Sicht sicher so gewollt, aber es wäre schön, wenn dem potenziellen Betreiber des Systems eine einfachere Möglichkeit als eine Veränderung des Codes geboten werden würde, um den Grad der Authentifizierung zu verändern. Gerade die öffentlich einsehbare Benutzerliste ist sehr kritisch und vereinfacht einem potenziellen Angreifer den Identitätsdiebstahl auf der Plattform, da dadurch sogar nur das Passwort erraten werden muss, nicht Benutzername und Passwort. Daher kann die Bewertung für dieses Kriterium nicht besser ausfallen als $\mathfrak{1} = \text{eher nicht erfüllt}$.

Integrity: In der nachfolgenden Tabelle sind die in den für diese Arbeit angelegten CKAN-Instanzen genutzten Subsysteme und die dazugehörigen bekannten Sicherheitslücken (nach CVE²) aufgeführt:

²Common Vulnerabilities and Exposures

Subsystem	Version	Systemtyp	Anzahl Sicherheitslücken	CVE-Typ
Docker	19.03.10	Virtualisierung	0 ¹	—
PostgreSQL	9.6.13	rel. Datenbank	2 [6]	Ausführung von Code
Apache Solr	6.6.2	Search Engine	4 [6]	Ausführung von Code
Redis	6.0.3	Key-Value-Store	0 ¹	

1: Die Version ist noch zu neu, sodass bestehende Sicherheitslücken aus vorherigen Versionen noch nicht bestätigt wurden.

Tabelle 6.8: CKAN: Integrity

Die Anzahl an bekannten Sicherheitslücken ist für alle vier untersuchten Subsysteme einstellig und damit vergleichsweise niedrig. Auch ist der CVSS³-Score der einzelnen Sicherheitslücken eher niedrig, außer bei zweien:

1. Eine der bekannten Sicherheitslücken von Apache Solr 6.6.2 mache es möglich, ein Skript in einem Parameter zu übermitteln, welches dann serverseitig ausgeführt werde. [6]
2. In PostgreSQL 9.6.13 könne ein Nicht-Superuser auf nicht genauer spezifizierte Weise Code mit den Berechtigungen eines Superusers ausführen. [6]

Der Sicherheitslücke von Apache Solr kann entgegengewirkt werden, indem der Port nach außen geschlossen bleibt; eventuell wird dadurch die Nutzung der API von externen Systemen eingeschränkt, sofern bestimmte API-Calls Solr im Hintergrund für die Suche verwenden. Dies geht jedoch nicht aus der Dokumentation hervor.

Damit ein potenzieller Angreifer sich die Sicherheitslücke von PostgreSQL zunutze machen kann, benötigt er einen Benutzeraccount für die Datenbank. Da es im Fall von CKAN - zumindest ohne den Einsatz von Extensions - nicht nötig ist, individuelle Accounts für die Datenbank zu vergeben, kommt das System mit dem Account aus, der während der Installation erstellt wird. Da dieser Account ohnehin ein Superuser-Account ist, wäre das Problem bei erfolgreichem Erlangen des Accounts durch einen potenziellen Angreifer freilich größer als das Ausnutzen der Sicherheitslücke.

Im Großen und Ganzen lassen sich die bekannten Sicherheitsprobleme also vermeiden bzw. umgehen und es folgt eine Bewertung von $\mathcal{B} = \text{erfüllt}$.

Non-repudiation: Im CKAN-Logfile, welches über den Docker-Befehl “docker-compose logs ckan” abgerufen werden kann, finden sich leider keine Angaben bezüglich durch

³Common Vulnerability Scoring System

Benutzer durchgeführte Aktionen. Dafür gibt es den sogenannten “Activity Stream” zu jedem Benutzer und Dataset sowie zu jeder Organization und Group. Der Activity Stream ist direkt in der Weboberfläche einsehbar und zeigt taggenau an, welche Aktionen an oder mit dem Objekt durchgeführt wurden, aber nicht alle Aktionen werden genau getrackt:

Aktion	Eintrag im Activity Stream
Erstellen eines Datasets	ja
Bearbeiten eines Datasets	ja ¹
Löschen eines Datasets	ja ²
Hinzufügen einer Datei zu einem Dataset	ja
Entfernen einer Datei aus einem Dataset	nein ³
Erstellen einer Organization	ja
Bearbeiten einer Organization	ja ¹
Löschen einer Organization	nein ²
Hinzufügen eines Benutzers zur Organization	nein
Entfernen eines Benutzers aus der Organization	nein
Erstellen einer Group	ja
Bearbeiten einer Group	ja ¹
Hinzufügen eines Datasets zu einer Group	nein
Entfernen eines Datasets aus einer Group	nein
Erstellen eines Benutzers	ja
Bearbeiten eines Benutzerprofils	ja ¹
Entfernen eines Benutzers	nein

1: keine Details zur Veränderung

2: gelöschte Objekte können aufgerufen werden, da sie bekanntermaßen nicht wirklich gelöscht werden

3: ein neuer Eintrag wird erzeugt, aber dort steht nur, dass das Dataset aktualisiert wurde

Tabelle 6.9: CKAN: Non-repudiation

Insgesamt werden für zehn der siebzehn in der Tabelle aufgeführten Aktionen, die sich in einem Event-Log als nützlich erweisen würden, Einträge in den jeweiligen Activity Streams erzeugt. Zudem wird nicht immer genau angegeben, welche Änderungen vorgenommen wurden und auch Uhrzeiten wären durchaus hilfreich. Diese Einträge werden in

der PostgreSQL-Datenbank des Systems gespeichert, aber die Tabellen enthalten nicht viel mehr hilfreiche Informationen. Daraus ergibt sich eine Bewertung von **1 = eher nicht erfüllt** für dieses Kriterium.

Authenticity: Ohne den Einsatz von Extensions gibt es in CKAN nur eine Anmeldeoption: Benutzername und Passwort. Im Grunde reicht dies für den Einsatz in den meisten Organisationen wohl aus, aber die HAW und auch die HCHS möchten das System auch für z.T. durch Geheimhaltungsvereinbarungen geschützte Daten verwenden. Das BSI⁴ empfiehlt inzwischen den Einsatz von Zweifaktor-Authentifizierung (2FA) bei jedem Dienst, der dies ermöglicht [4]. Vor allem bei sicherheitsrelevanten Themen, wozu geheimzuhaltende Daten gehören, ist der Einsatz von 2FA inzwischen unablässig. Daher wird CKAN für dieses Kriterium mit **1 = eher nicht erfüllt** bewertet.

Accountability: Wie schon beim Kriterium *Non-repudiation* ist für die Erfüllung dieses Kriteriums ein guter Event-Log vonnöten, also fällt die Bewertung für dieses Kriterium ebenso mit **1 = eher nicht erfüllt** aus.

6.2.1.7 Maintainability (Wartbarkeit)

Modularity:

1. *Wurden Interfaces definiert und ordnungsgemäß verwendet?*

CKAN wurde hauptsächlich in Python geschrieben. Python setzt auf gut funktionierende multiple Vererbung [23] und das Konzept des Duck-Typing [23] und kommt daher auch ohne Interfaces aus, das Konzept existiert in der Programmiersprache nicht. Die Meinungen gehen diesbezüglich auseinander, aber letztlich ist es eine subjektive Entscheidung, welche Art Programmiersprache von einer Person bevorzugt wird.

Eine stichprobenartige Überprüfung einiger Klassen hat ergeben, dass in diesem Projekt der Vererbungsmechanismus genutzt wird. Die Extending- & Contributing-Guides der Dokumentation enthalten eine Anweisung, den Python Style Guide von Google zu befolgen, welcher explizit zum Einsatz von Vererbung auffordert [26].

Interessanterweise findet sich in der Dokumentation von CKAN im Extending Guide ein Abschnitt namens “Plugin interfaces reference”, welchem sich entnehmen lässt, dass für das Projekt eine Klasse “Interface” geschrieben wurde, von der eine Reihe weiterer Klassen erben, welche durch das typische vorangestellte große “I”

⁴Bundesamt für Sicherheit in der Informationstechnik

als Interfaces kenntlich gemacht wurden. Diese “Interfaces” sollen das Erweitern von CKAN durch Plugins erleichtern, also erfüllen sie im Grunde den tatsächlichen Zweck des objektorientierten Konzepts eines Interfaces, obwohl die gewählte Programmiersprache sich implizit gegen dieses Konzept ausspricht.

2. Wird durch die Dokumentation und/oder Kommentare klar, welche Komponenten voneinander abhängen?

Die Dokumentation enthält im Contributing Guide einen Abschnitt “CKAN code architecture”. In diesem Abschnitt findet sich neben näheren Erklärungen ein Schaubild, welches grob wiedergibt, wie die einzelnen Komponenten untereinander agieren:

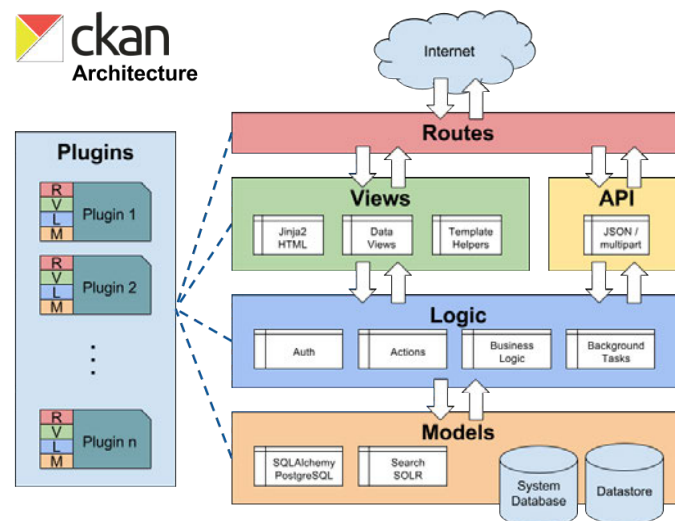


Abbildung 6.5: CKAN: Überblick über die Architektur [22]

Leider ist dies die feinste Granularitätsstufe, die die Dokumentation hinsichtlich der Architektur aufweist. Wenn sich jemand dazu entschließen sollte, sich an dem Projekt beteiligen zu wollen, ist er demnach gezwungen, sich große Teile des gut kommentierten Source Codes anzusehen, um die Systemstruktur zu verstehen. Dies gilt allerdings nicht für die Erweiterung des Systems durch Extensions, da die Extending- & API-Guides in der Dokumentation sehr umfangreich und gut verständlich geschrieben sind.

Trotz der etwas fragwürdigen Entscheidung Python zu benutzen, aber dann nicht ganz ohne Interfaces auskommen zu wollen und der nur grob dargestellten Systemarchitektur ist dank des gut kommentierten und recht überschaubaren Codes mit etwas Zeitaufwand einigermaßen gut nachzuvollziehen, wie der Code funktioniert, was wichtig für die Modularität des Codes ist. Daher fällt die Bewertung mit $2 = \textit{eher erfüllt}$ aus.

Reusability: Viele Systemfunktionen können schon mithilfe der großzügig ausfallenden API verwendet werden. Die im Kriterium *Modularity* schon erwähnten nicht ganz Python-konformen Interfaces und gut kommentierter Code weisen dem wiederverwendungsfreudigen Entwickler einen recht komfortablen Weg, mit wenig eigenem Code zur gewünschten Veränderung zu gelangen. Die Bewertung für dieses Kriterium fällt daher mit $3 = \textit{erfüllt}$ aus.

Analysability: Wie in *Modularity* schon angesprochen, wird die Systemarchitektur in der Dokumentation nur grob angerissen und es existieren keine Komponentendiagramme mit mehreren Ebenen oder ähnliche Darstellungen, die bei umfangreichen Projekten wünschenswert wären. Der Code ist zwar gut kommentiert und recht übersichtlich gehalten, aber im Grunde sollte ein Softwareprojekt auch anhand der Dokumentation zu einem gewissen Grad analysierbar sein, was hier leider nicht gegeben ist. Daher wird CKAN für dieses Kriterium mit $1 = \textit{eher nicht erfüllt}$ bewertet.

Modifiability: Bei den im Rahmen dieser Arbeit durchgeführten Modifizierungen des Codes in Form eines Prototyps einer Extension ist aufgefallen, dass die ausführliche Dokumentation z.T. falsch oder nicht mehr aktuell ist. Die Veränderung der Benutzeroberfläche erfolgt z.B. mithilfe der Web-Template-Engine Jinja2, die es ermöglicht, Python-Ausdrücke in einer HTML-Datei zu verwenden. Durch die Verwendung dieser Template-Engine wirkt die Programmierung der Benutzeroberfläche baukastenähnlich, was zu schnellen Erfolgserlebnissen führt. Da die Modifizierung CKAN-seitig fast fehlerlos verlief, folgt eine Bewertung von $2 = \textit{eher erfüllt}$.

6.2.1.8 Portability (Portierbarkeit)

Adaptability: CKAN wurde im Rahmen der Arbeit für beide Anwendungsfälle auf unterschiedlichen Systemen mithilfe des offiziellen CKAN-Docker-Images und der offiziellen Docker-Installationsanleitung für CKAN in Docker-Containern installiert und gehostet. Docker ist ein Service für Containervirtualisierung und bietet den Vorteil, dass alle in

Docker-Containern laufenden Softwareinstanzen grundsätzlich auf allen Betriebssystemen, die von Docker unterstützt werden, ebenfalls laufen. Docker ist für Windows-, Mac- und Linux-Betriebssysteme erhältlich. Im Rahmen der Arbeit wurde CKAN auf virtuellen und physikalischen Maschinen mit unterschiedlichen Ubuntu-Versionen (18.04, 19.10, 20.04) und unterschiedlichen Hardware-Spezifikationen (zwei oder vier Kerne, 4 oder 8 GB RAM) betrieben und lief in jeder Version problemlos.

Da die Entwicklung einer Extension in einem Docker-Container auf Grund dessen kompliziert und unkomfortabel ist, dass die in den Containern liegenden Dateien nur mittels CLI⁵ und nicht etwa in einer Entwicklungsumgebung mit Code Highlighting und Linting bearbeitet werden können, wurde zudem eine CKAN-Instanz ohne Docker auf einem System mit Ubuntu 16.04 installiert, da dies für die “Installation from Source” in der Installationsanleitung so vorgesehen war. Auch diese Instanz lief reibungslos, somit folgt eine Bewertung von $\mathfrak{Z} = \text{erfüllt}$.

Installability: Die Installation des Systems wurde auf einer Reihe von Systemen durchgeführt. In der folgenden Tabelle finden sich alle Kombinationen aus Betriebssystem und CKAN-Version wieder, die ausprobiert wurden:

Betriebssystem	CKAN-Version	Installation problemlos?
Ubuntu 16.04	2.8.2	ja
Ubuntu 18.04	2.8.2	nein ¹
Ubuntu 18.04	2.8.2 in Docker	ja
Ubuntu 20.04	2.8.2 in Docker	ja

1: Die verwendete offizielle Installationsanleitung bezog sich ausdrücklich auf die Ubuntu-Versionen 14.04 und 16.04.

Tabelle 6.10: CKAN: Installability

Insgesamt funktionierte die Installation also in drei von vier Versuchen reibungslos, wobei bei dem nicht erfolgreichen Versuch die Betriebssystem-Version in der Anleitung missachtet wurde, also wird das Kriterium mit $\mathfrak{Z} = \text{erfüllt}$ bewertet.

Replaceability: In funktionaler Hinsicht ist CKAN in der Lage Dataverse vollständig zu ersetzen, wie ein Vergleich der beiden Tabellen 6.4 und 6.11 zeigt. Daraus folgt eine Bewertung von $\mathfrak{Z} = \text{erfüllt}$.

⁵Command Line Interface

6.2.2 Dataverse

In diesem Abschnitt erfolgt die Bewertung des Systems Dataverse. Genau wie bei der Bewertung von CKAN gab es dank der vorher genau definierten Vorgehensweise keine Probleme.

6.2.2.1 Functional Suitability (Funktionelle Eignung)

Functional Completeness: Um die funktionelle Vollständigkeit zu überprüfen, wird nachfolgend für jedes Ziel und jede Aufgabe, das bzw. die in 3.1.3 definiert wurden, beschrieben, inwiefern es/sie von Dataverse erfüllt wird.

[Z01] Abbildung der Organisationshierarchie im Berechtigungssystem:

Das Berechtigungssystem von Dataverse ist etwas anders aufgebaut als bei CKAN. Berechtigungen werden pro Dataverse in Form von Rollen vergeben. Hierfür ist es wichtig noch einmal hervorzuheben, dass Dataverses weitere Dataverses enthalten können. Wenn ein Benutzer eine bestimmte Rolle im Root-Dataverse erhält, hat er diese Rolle auch in allen Kind-Dataverses inne. In der Standardinstallation gibt es bereits acht verschiedene Rollen:

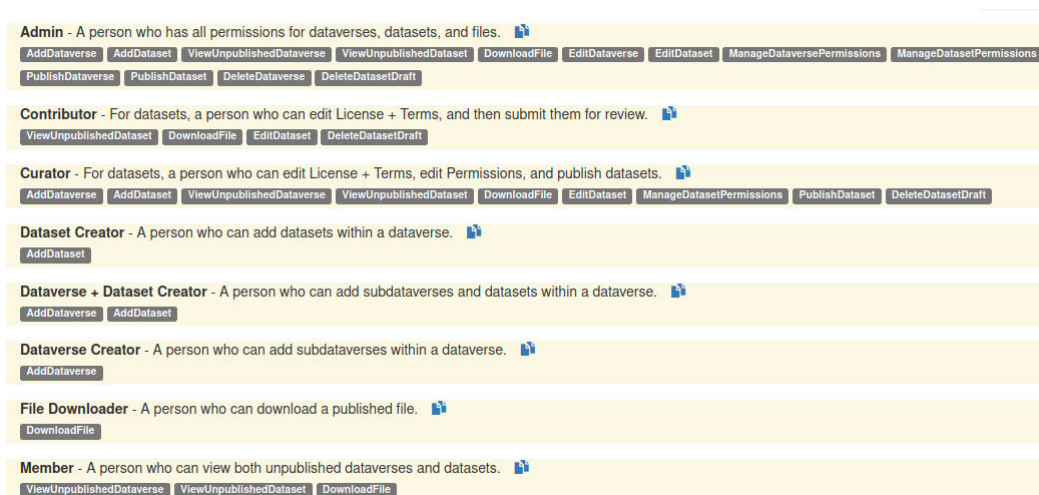


Abbildung 6.6: Dataverse: die acht Standard-Benutzerrollen

Wie in Abbildung 6.6 zu sehen ist, sind jeder Rolle gewisse Funktionen zugewiesen (graue Kästchen unterhalb der Rollenbeschreibung). Über die Schaltfläche “Add New Role”, können auf einfache Art und Weise benutzerdefinierte Rollen hinzugefügt werden. Hierbei werden die Funktionen per Checkbox-Auswahl zugewiesen. Durch diese Anpassbarkeit können sämtliche vorstellbaren Organisationsstrukturen abgebildet werden.

Jeder Benutzer, der in einem Dataverse keine zugewiesene Rolle innehat, kann innerhalb dieses Dataverses nur veröffentlichte Datasets und Kind-Dataverses einsehen. Das Ziel wird als erfüllt angesehen.

[Z02] Zugriff auf bestimmte Daten/Dateien/Projekte nur durch bestimmte Nutzer:

Jeder Besucher einer Dataverse-Seite kann jedes veröffentlichte Dataverse und darin veröffentlichte Datasets einsehen, unabhängig davon, ob er registriert ist oder nicht. Jedes Dataverse und auch jedes einzelne Dataset im Dataverse kann von dazu berechtigten Mitgliedern des Dataverses auch auf “Unpublished” gestellt werden. In diesem Fall ist das Dataverse/Dataset nur für registrierte Nutzer, die ebenfalls Mitglieder des Dataverses sind und die entsprechenden Berechtigungen haben, sichtbar. Das Ziel wird als erfüllt angesehen.

[Z03] geordnete Sammlung aller erhobenen Forschungsdaten:

Da die Berechtigungen zum Hochladen Dataverse-weise vergeben wird, kann genau gesteuert werden, welcher Benutzer in welchem Dataverse seine Datasets hochlädt. Auch hier böte es sich an, von den Nutzern zu verlangen, dass für ein Dataverse eine wohlüberlegte Kategorie gewählt und eine aussagekräftige Beschreibung für jedes hochgeladene Objekt hinterlegt wird. Das Ziel wird als erfüllt angesehen.

[Z04] Kategorisierung von Projekten (Tagging):

Dataverses können jeweils genau einer Kategorie zugewiesen werden, während die darin enthaltenen Datasets eine beliebige Anzahl an Keywords erhalten kann, nach denen auch gesucht werden kann. Das Ziel wird als erfüllt angesehen.

[Z05] Zuordnung von schriftlichen Ergebnissen zu Daten und Dateien:

Genau wie bei CKAN kann ein Dataset mehrere Dateien in allen möglichen Formaten enthalten. Somit können auch schriftliche Ergebnisse in Form von Textdateien den Daten beigefügt werden. Das Ziel wird als erfüllt angesehen.

[Z06] Teilen von Forschungsdaten mit anderen Forschern:

Auch dieses Ziel wird von Dataverse ähnlich wie von CKAN erfüllt. Veröffentlichte Datensets können von allen Besuchern der Seite verwendet werden. Wenn die Daten nur mit bestimmten Nutzern geteilt werden sollen, müssen diese Nutzer sich registrieren und dem Dataverse als Mitglied hinzugefügt werden. Dann können sie auch unveröffentlichte Datensets einsehen. Das Ziel wird als erfüllt angesehen.

[Z07] Abbildung eines Variablenkatalogs unabhängig von tatsächlich hochgeladenen Dateien (nur HCHS):

Auch von Dataverse wird dieses Ziel nicht erfüllt. Hier ist nicht einmal eine unschöne Lösung wie bei CKAN möglich, da in Dataverse keine Previews für Tabellendateien existieren, mit denen der Variablenkatalog online angesehen werden könnte. Das Ziel wird als nicht erfüllt angesehen.

[A01] Upload vieler Dateien in einem Projekt:

Ein Dataset ist in der Anzahl der zugehörigen Dateien nicht begrenzt. Der Upload mehrerer Dateien gleichzeitig kann sowohl per Webinterface als auch mithilfe der API erfolgen. Die Aufgabe wird als erfüllbar angesehen.

[A02] Upload von Dateien bis 10GB Größe:

Eine Obergrenze für die Dateigröße kann konfiguriert werden, aber ist standardmäßig deaktiviert. Die Aufgabe wird als erfüllbar angesehen.

[A03] Suchen und Finden von Dateien:

Dateien können mithilfe der Suchleiste durch Eingabe eines Teils des Namens oder des Dateiformats gefunden werden. Zudem gibt es eine erweiterte Suche, in der über 30 weitere Suchparameter eingestellt werden können. Die Aufgabe wird als erfüllbar angesehen.

[A04] Suchen und Finden von Projekten:

Für Datasets gilt dasselbe wie für Dateien. Die Aufgabe wird als erfüllbar angesehen.

[A05] Suchen und Finden von Benutzergruppierungen:

Für Dataverses gilt dasselbe wie für Dateien und Datasets. Die Aufgabe wird als erfüllbar angesehen.

[A06] Suchen und Finden von einzelnen Benutzern:

Die Eingabe eines Benutzernamens in der Suchleiste erzielt keine Treffer. Die Suche nach Vor- oder Nachnamen funktioniert, wenn der gesuchte Name als Autor eines Datasets oder Dataverses angegeben ist. Da das entsprechende Feld ein automatisch ausgefülltes Pflichtfeld ist, sollte dies im Regelfall zutreffen. Administratoren können zudem über das Dashboard alle registrierten Benutzer anhand von Benutzernamen, Vor- und Nachnamen sowie Email-Adresse suchen und finden. Die Aufgabe wird als erfüllbar angesehen.

[A07] Download von Dateien mit entsprechender Autorisierung:

Genau wie bei CKAN kann grundsätzlich erst einmal jeder Besucher der Seite veröffentlichte Dateien herunterladen. Ebenso wie bei CKAN können Datasets unveröffentlicht bleiben. In diesem Fall können nur Mitglieder des entsprechenden Dataverses mit entsprechenden Berechtigungen die Dateien einsehen. Die Aufgabe wird als erfüllbar angesehen.

[A08] Verwalten von einzelnen Benutzern:

Über die Startseite eines Dataverses können per “Edit”-Schaltfläche Benutzer dem Dataverse hinzugefügt und ihnen Rollen zugewiesen oder genommen werden. Zudem können Administratoren über das Dashboard Benutzern alle Rollen in allen Dataverses entziehen oder sie zu Superusern machen. Allerdings gibt es keine Möglichkeit, in der Weboberfläche einen Benutzer zu entfernen. Daher wird die Aufgabe als nicht vollständig erfüllbar angesehen.

[A09] Verwalten von Benutzern in Gruppen:

Anstatt einzelnen Benutzern Rollen in Dataverses zuzuweisen, kann man die Benutzer auch in einer Gruppe sammeln und dann der Gruppe die entsprechenden Rollen zuweisen. Aber da auch hierbei die Funktion “Benutzer entfernen” fehlt, wird die Aufgabe als nicht vollständig erfüllbar angesehen.

[A10] Veröffentlichen von Forschungsergebnissen:

Sobald ein Dataset auf “public” gesetzt wird, ist es von jedem Besucher der Hauptseite in der Suche auffindbar. Also kann jeder Benutzer, der seine Ergebnisse veröffentlichen möchte, einfach seine zugehörigen Datasets in seiner Arbeit erwähnen und interessierte Leser mithilfe eines Links zu den zugehörigen Dateien führen. Zudem kommuniziert Dataverse mit der Plattform “DataCite”, wo der DOI des Datasets hinterlegt wird. Die Aufgabe wird als erfüllbar angesehen.

[A11] Bearbeiten des Online-Variablenkatalogs im Browser (nur HCHS):

Da in der Standard-Dataverse-Installation die Abbildung eines Online-Variablenkatalogs nicht möglich ist (s. [Z07]), ist das Bearbeiten des Katalogs im Browser folglich auch nicht möglich.

Die Erkenntnisse noch einmal kurz in einer Tabelle zusammengefasst:

Ziel/Aufgabe	wird von Dataverse erfüllt
Z01	Ja
Z02	Ja
Z03	Ja
Z04	Ja
Z05	Ja
Z06	Ja
Z07	Nein
A01	Ja
A02	Ja
A03	Ja
A04	Ja
A05	Ja
A06	Ja
A07	Ja
A08	Nein
A09	Nein
A10	Ja
A11	Nein

Tabelle 6.11: Dataverse: Functional Completeness

Aus dem Ergebnis 14x ja und 4x nein ($\frac{14}{18} \approx 0.78$) ergibt sich nach Tabelle 6.2 eine Bewertung von $\mathcal{Q} = \textit{eher erfüllt}$.

6.2.2.2 Performance Efficiency (Performanz)

Time Behaviour: Alle Ladezeiten auf der Plattform bleiben regelmäßig unter einer Sekunde, mit Ausnahme vom Dateiupload bei größeren Dateien, was wie bei CKAN auch abhängig von der Upload-Rate der genutzten Internetverbindung ist. Im Gegensatz zu CKAN gibt es beim Upload in Dataverse eine Fortschrittsanzeige. Auch Dataverse bietet eine API-Funktion für die Erstellung von Datasets und den Upload von Dateien an. Das Kriterium wird mit $\mathcal{Z} = \text{erfüllt}$ bewertet.

Resource Utilization: Wie auch bei CKAN wurde die Auslastung mithilfe des Tools “htop” erfasst, indem das Attribut “Load Average” ausgewertet wurde. In der folgenden Tabelle wird aufgeführt, wie hoch die CPU- und RAM-Auslastungen während typischer Aufgaben auf dem Testsystem (Ubuntu 18.04, 4GB RAM, 2 CPU-Kerne) liegen.

Aufgabe	CPU	RAM
Leerlauf ohne Dataverse	4%	6.3%
Leerlauf inkl. Dataverse im Leerlauf	10.5%	40.4%
Upload einer großen Datei	13.5%	41.1%
mehrere gleichzeitige Suchabfragen ¹	29%	42.2%
mehrere gleichzeitige Downloads ¹	15%	42.4%

1: Diese Aufgabe wurde mithilfe der API simuliert

Tabelle 6.12: Dataverse: Resource Utilization

Bei allen Belastungstests blieben die Auslastungen der CPU und des RAM weit unter 80%, daraus ergibt sich eine Bewertung von $\mathcal{Z} = \text{erfüllt}$.

Capacity: Die maximale Anzahl von Dateien sowie die maximale Dateigröße sind unlimitiert. Daraus ergibt sich eine Bewertung von $\mathcal{Z} = \text{erfüllt}$.

6.2.2.3 Compatibility (Kompatibilität)

Co-existence: Auch auf der virtuellen Maschine, die die Dataverse-Instanz der HAW beheimatet, wurde ein LAMP-Stack installiert und auf Port 80 läuft eine Website. Auch hier gilt: Wenn die Ports konfiguriert werden können, können auch verschiedene Netzwerkanwendungen nebeneinander existieren. Daraus ergibt sich eine Bewertung von $\mathcal{Z} = \text{erfüllt}$.

Interoperability: Der Dataverse-API-Guide ist in fünf Abschnitte gegliedert:

1. Die Search API ermöglicht das Suchen nach Dataverses, Datasets und Dateien.
2. Mit der Data Access API können Dateien einzeln oder gebündelt heruntergeladen werden.
3. Die Native API ermöglicht das Anwenden fast aller Funktionen, die auch mit der GUI möglich sind - und sogar mehr. Von der Erzeugung aller möglichen Objekte wie Dataverses, Datasets, Gruppen und Rollen über das Editieren von Datasets und Dateien bis hin zur Benutzerverwaltung und sogar der in der GUI fehlenden Funktion, einen Benutzer zu löschen ist alles möglich.
4. Mithilfe der Metrics API können Statistiken zu der Dataverse-Installation abgerufen werden.
5. Die SWORD⁶ API bietet Funktionen rund um die Manipulation von Datasets an (z.B. Erzeugen, Abrufen und Publizieren). Im Gegensatz zu den ebenbürtigen Funktionen aus der Data Access und Native API werden die Ergebnisse in XML- und nicht in JSON-Form wiedergegeben.

Im Grunde ist es möglich, alle vorstellbaren Aufgaben per API-Call zu erledigen. Daher ergibt sich eine Bewertung von $\mathfrak{3} = \text{erfüllt}$.

6.2.2.4 Usability (Benutzbarkeit)

Appropriateness Recognizability: Der Autor konnte alle wichtigen Systemfunktionen auf Anhieb verstehen. Einzig das Fehlen einer "Benutzer löschen"-Funktion in der Weboberfläche fällt negativ auf und schränkt einen potenziellen Systemadministrator ggf. ein, wenn er keine Erfahrung mit der Einsatz von APIs hat. Daher kann die Bewertung nicht mit Höchstpunktzahl erfolgen und es ergibt sich eine Bewertung von $\mathfrak{2} = \text{eher erfüllt}$.

Learnability: Der Autor hatte keine Probleme damit, die Systemfunktionen zu verstehen. Die Bewertung erfolgt daher mit $\mathfrak{3} = \text{erfüllt}$.

Operability: In der nachfolgenden Tabelle steht für jede der wahrscheinlich am häufigsten verwendeten Dataverse-Funktionen das Ergebnis der Überprüfung, ob sie in drei Schritten erfüllbar ist. Dabei wird jeweils davon ausgegangen, dass der Benutzer bereits

⁶Simple Web-service Offering Repository Deposit

zu einer für die Aufgabe angemessenen Seite navigiert ist (z.B. beim ‘‘Hinzufügen eines Benutzers zu einem Dataverse’’ auf der Hauptseite des gewünschten Dataverses).

Funktionalität	≤ 3 Schritte
Suchen und Finden eines Datasets	ja
Suchen und Finden eines Dataverses	ja
Suchen und Finden einer Group	nein
Suchen und Finden eines Benutzers	ja ¹
Anlegen eines neuen Datasets	ja
Anlegen eines neuen Dataverses	ja
Anlegen einer neuen Group	nein
Anlegen eines neuen Benutzers	nein ²
Hinzufügen einer Datei zu einem bestehenden Dataset	ja
Entfernen einer Datei aus einem Dataset	ja
Bearbeiten eines Datasets	ja
Löschen eines Datasets	ja
Hinzufügen eines Benutzers zu einem Dataverse	nein
Entfernen eines Benutzers aus einem Dataverse	ja
Zurücksetzen eines Benutzerkennworts	nein ³
Zurücksetzen eines API-Keys	ja
Previewen einer Datei	nein ⁴
Herunterladen einer Datei	ja

1 Im Gegensatz zu CKAN haben in Dataverse Benutzer keine eigenen Seiten, auf denen sie Informationen über sich oder ihre Forschungsarbeit preisgeben können. Das Ergebnis bezieht sich daher auf von Benutzern freigegebene Dataverses, Datasets und Dateien.

2 Das Anlegen eines neuen Benutzers ist für einen Administrator in der Weboberfläche nicht möglich. Die Registrierung eines neuen Nutzers ist grundsätzlich in drei Schritten möglich, aber hierbei geht es um die Erstellung eines Accounts durch einen Administrator für einen Dritten.

3 Das Zurücksetzen eines Benutzerkennworts für einen Dritten durch einen Administrator ist nicht möglich. Es gibt eine Funktion zum Zurücksetzen des Passworts auf der Loginseite, aber in der installierten Instanz funktioniert diese Funktion nicht.

4 In der Standardinstallation kann Dataverse keine Dateien im Browser darstellen.

Tabelle 6.13: Dataverse: Operability

Aus dem Ergebnis 12x ja und 6x nein ($\frac{12}{18} \approx 0.67$) ergibt sich nach Tabelle 6.2 eine Bewertung von $2 = \text{eher erfüllt}$.

User Error Protection: Wie bei CKAN auch wird beim Einsatz kritischer Operationen wie dem Löschen von Datasets mit einem Pop-up abgefragt, ob der Benutzer dies wirklich tun möchte. Im Gegensatz zu CKAN werden hier beim Löschen allerdings wirklich Einträge entfernt und nicht nur “unsichtbar gemacht”. Eine kurze Suche in einer Logtabelle (“actionlogrecord”) und ein Abgleich der ID eines gerade gelöschten Datasets mit der Tabelle “dataset” in der Dataverse-Datenbank bestätigt, dass ein gerade gelöscht Dataset keinen Eintrag mehr in der Datenbank aufweist. Dass es hier keine Möglichkeit gibt, eine gelöschte Entität wiederherzustellen, kann kritisch sein. Daher erfolgt eine Bewertung mit $1 = \text{eher nicht erfüllt}$.

User Interface Aesthetics: Es werden dieselben drei Eigenschaften wie bei CKAN überprüft:

1. *Der Text muss gut lesbar sein.* Es werden hauptsächlich blaue und schwarze Schrift auf weißem Hintergrund verwendet, die Schriftgröße ist gut gewählt. Je nach Bildschirm und Bildschirmhelligkeit kann ein komplett weißer Hintergrund unangenehm sein, weshalb ein helles oder noch besser dunkles Grau für den Hintergrund zu bevorzugen wäre. Auch Dataverse glänzt mit Responsive Design, das die mobile Verwendung angenehm macht.

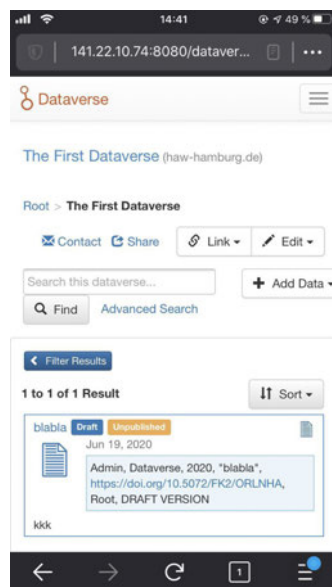


Abbildung 6.7: Dataverse: Responsive Design ermöglicht Verwendung an Mobilgeräten

2. *Schaltflächen müssen eindeutige Funktionen erfüllen.* Hier gibt es keine Gründe zur Beanstandung. Alle Schaltflächen sagen äußerst eindeutig aus, was sie tun. Dataverse verstärkt die Eindeutigkeit der Schaltflächen mithilfe von Icons, die z.T. zusätzlich zu einer Beschriftung vorhanden sind, anstatt wie CKAN in der Menüleiste ausschließlich Icons zu verwenden.
3. *Der Benutzer darf beim Umgang mit der Software keinen Stress empfinden.* Wie bei der ersten Eigenschaft schon erwähnt, empfindet der Autor den reinweißen Hintergrund als etwas unangenehm. Darüber hinaus werden keine Signalfarben für Buttons mit kritischen Optionen verwendet. Insgesamt können durch diese beiden Punkte bei längerem Arbeiten womöglich eher Fehler passieren.

Da die Benutzeroberfläche dem Autor nicht besonders zusagt und durch Theming nur der Header einer Seite angepasst werden kann, ist die Höchstpunktzahl ausgeschlossen und Dataverse erhält für dieses Kriterium eine Bewertung von $\mathcal{2} = \text{eher erfüllt}$.

Accessibility: Die Benutzeroberfläche im Web ist sehr simpel zu bedienen. Im Gegensatz zu CKAN gibt es in der Standardinstallation keine Option, die Sprache benutzergebunden einzustellen. Dies kann aber nachträglich über eine Property-Datei aktiviert werden. Ebenso wie bei CKAN sollten Systemadministratoren etwas Erfahrung im Umgang mit Docker und Command-Line-Interfaces haben und sich mit einer Linux-Umgebung auskennen. Die anderen beiden erwähnten Benutzergruppen (Studierende und Wissenschaftler) müssen sich auch hier nur mit der Weboberfläche auseinandersetzen können, wozu keine besonderen Fähigkeiten vonnöten sind. Daraus ergibt sich eine Bewertung von $\mathcal{3} = \text{erfüllt}$.

6.2.2.5 Reliability (Zuverlässigkeit)

Maturity: Genau wie bei CKAN gab es in der gesamten Nutzungszeit keinen einzigen Fehler. Daher fällt die Bewertung auch hier mit $\mathcal{3} = \text{erfüllt}$ aus.

Availability: Auch was die Systemverfügbarkeit angeht, gab es keinerlei Probleme, weshalb das Kriterium ebenfalls mit $\mathcal{3} = \text{erfüllt}$ bewertet wird.

Recoverability: Das Herunterfahren der Docker-Container während der Erstellung eines neuen Dataverses oder Datasets führt dazu, dass das Formular erneut ausgefüllt werden muss, da der Benutzer nach erneutem Hochfahren der Container auf die Startseite weitergeleitet wird und sich erneut einloggen muss. Zudem werden gelöschte Daten tatsächlich

unwiederbringlich gelöscht. In Kombination führt dies zu einer Bewertung von $0 = \textit{nicht erfüllt}$.

6.2.2.6 Security (Sicherheit)

Confidentiality: Wie bei CKAN kann jeder Besucher der Website alle veröffentlichten Dataverses und dazugehörige veröffentlichte Datasets einsehen. Anders als bei CKAN werden bei Dataverse aber keine Benutzernamen preisgegeben, überhaupt sind diese nur für Administratoren einsehbar, was eine Angriffsfläche weniger bedeutet. In der Dokumentation finden sich keine Informationen darüber, ob die “Registrieren”-Funktion sich ohne Veränderung des Codes deaktivieren lässt, somit lässt sich nicht kontrollieren, wer sich registrieren kann. Dies ist jedoch nicht weiter schlimm, da ein registrierter Benutzer ohne zugewiesene Berechtigungen nicht mehr machen kann, als ein nicht registrierter Besucher der Webseite.

Die Authentifizierung kann sowohl per Benutzername/Passwort als auch über verschiedene Anbieter wie Google, Github und Microsoft Azure AD, die allesamt das OAuth2-Protokoll verwenden, erfolgen. Während für die native Benutzername/Passwort-Variante keine Zweifaktor-Authentifizierung verfügbar ist, bieten die o.g. Anbieter dies an, was für zusätzliche Sicherheit sorgen kann.

Genau wie bei CKAN können nur Nutzer unveröffentlichte Datasets sehen, die Mitglieder mit bestimmten Berechtigungen des Dataverses, welches das Dataset enthält, sind.

Insgesamt wirkt Dataverse besser gegen Fremdzugriff geschützt als CKAN, aber eine deaktivierbare “Registrieren”-Funktion wäre wünschenswert. Daher wird das Kriterium mit $2 = \textit{eher erfüllt}$ bewertet.

Integrity: In der nachfolgenden Tabelle sind die in den für diese Arbeit angelegten Dataverse-Instanzen genutzten Subsysteme und die dazugehörigen bekannten Sicherheitslücken (nach CVE) aufgeführt:

Subsystem	Version	Systemtyp	Anzahl Sicherheitslücken	CVE-Typ
Docker	19.03.11	Virtualisierung	0 ¹	—
PostgreSQL	9.6.17	rel. Datenbank	0 ¹	—
Apache Solr	7.3.0	Search Engine	3 [6]	Ausführung von Code
Glassfish	4.1	Anwendungsserver	1 [6]	Directory Traversal ²

1: Für diese Version wurden keine CVEs gefunden.

2: Zugriff auf Dateien und Verzeichnisse durch Eingabe von URLs möglich[10]

Tabelle 6.14: Dataverse: Integrity

Die Anzahl an bekannten Sicherheitslücken ist auch bei Dataverse für alle vier untersuchten Subsysteme einstellig und damit vergleichsweise niedrig. Auch ist der CVSS-Score der einzelnen Sicherheitslücken eher niedrig, außer bei zweien:

1. Die Sicherheitslücke, die es in Apache Solr Version 6.6.2 ermöglichen soll, ein Skript in einem Parameter zu übermitteln, welches dann serverseitig ausgeführt wird, existiere auch in Version 7.3.0 noch. [6]
2. Das Admin-Portal von Glassfish 4.1 ermögliche ein Directory Traversal, wodurch unerlaubter Zugriff auf Dateien auf dem Server möglich sei. [6]

Wie in 6.2.1.6 erwähnt, ist die erste Sicherheitslücke durch Schließen des von Solr genutzten Ports lösbar, wodurch evtl. API-Funktionalitäten verloren gehen können. Das Admin-Portal von Glassfish muss zum Verwenden von Dataverse nicht einmal aufgerufen werden, also könnte man hier ebenfalls einfach den Port nach außen verschließen, um das Problem zu beheben. Alles in allem sind die Komponenten nicht weniger sicher als die von CKAN, daher erfolgt auch hier eine Bewertung von $\mathcal{3} = erfüllt$.

Non-repudiation: Dataverse bietet zwar in der Weboberfläche kein Pendant zu CKANs “Activity Stream” an, aber in der PostgreSQL-Datenbank einer Dataverse-Instanz existiert die Tabelle “actionlogrecord”, welche nützliche Informationen enthält, wie der nachfolgende Screenshot eines CSV-Exports der Tabelle zeigt:

actionsubtype	info	starttime	useridentifier
login		2020-06-18 15:21:31.361	@test
logout		2020-06-18 15:21:37.475	@test
login		2020-06-18 15:21:45.183	@dataverseAdmin
logout		2020-06-18 15:22:34.739	@dataverseAdmin
passwordResetRequest	Email Address: pascal.heinsohn9194@gmail.com	2020-06-18 15:22:48.16	
passwordResetSent	Email Address: pascal.heinsohn9194@gmail.com	2020-06-18 15:22:48.187	
login		2020-06-18 15:24:19.208	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.GetPrivateUriCommand	[3 The First Dataset]	2020-06-18 15:24:26.242	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.UpdateDatasetVersionCommand	[3 The First Dataset]	2020-06-18 15:24:40.386	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.GetPrivateUriCommand	[3 The First Dataset]	2020-06-18 15:24:41.772	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.DeleteDataFileCommand	[3 The First Dataset] DataFile:4	2020-06-18 15:25:11.683	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.UpdateDatasetVersionCommand	[3 The First Dataset]	2020-06-18 15:25:11.611	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.GetPrivateUriCommand	[3 The First Dataset]	2020-06-18 15:25:12.991	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.GetPrivateUriCommand	[3 The First Dataset]	2020-06-18 15:25:29.766	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.GetPrivateUriCommand	[3 The First Dataset]	2020-06-18 15:25:57.621	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.AssignRoleCommand	[AuthenticatedUser identifier:@test] has been given DataverseRole[id=7, #	2020-06-18 15:26:55.067	@dataverseAdmin
generateApiToken	user:@dataverseAdmin token:b3610f53-ebce-4fd7-9c0f-ec5fb8a8a0e2	2020-06-18 15:30:35.074	
edu.harvard.iq.dataverse.engine.command.impl.GetPrivateUriCommand	[5 The Second Dataset]	2020-06-18 15:31:18.663	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.GetPrivateUriCommand	[5 The Second Dataset]	2020-06-18 15:32:18.511	@dataverseAdmin
login		2020-06-19 11:39:50.248	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.GetPrivateUriCommand	[3 The First Dataset]	2020-06-19 11:40:02.187	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.DeleteDataFileCommand	[3 The First Dataset] DataFile:13	2020-06-19 11:40:18.625	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.DestroyDatasetCommand	[3 The First Dataset]	2020-06-19 11:40:18.609	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.GetPrivateUriCommand	[5 The Second Dataset]	2020-06-19 11:40:56.278	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.DeleteDataFileCommand	[5 The Second Dataset] DataFile:6	2020-06-19 11:41:12.988	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.DeleteDataFileCommand	[5 The Second Dataset] DataFile:8	2020-06-19 11:41:13.065	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.DeleteDataFileCommand	[5 The Second Dataset] DataFile:9	2020-06-19 11:41:13.12	@dataverseAdmin
edu.harvard.iq.dataverse.engine.command.impl.DeleteDataFileCommand	[5 The Second Dataset] DataFile:10	2020-06-19 11:41:13.175	@dataverseAdmin

Abbildung 6.8: Dataverse: gekürzte Log-Tabelle “actionlogrecord”

Die vier hier abgebildeten Spalten enthalten gemeinsam viele nützliche Informationen, die benötigt werden, um nachzuvollziehen, welcher Benutzer wann welche Aktion durchgeführt hat. In der folgenden Tabelle wird dargestellt, welche der zu loggenden Aktionen in der Log-Tabelle wiederzufinden sind:

Aktion	Eintrag in der Log-Tabelle
Erstellen eines Datasets	ja
Bearbeiten eines Datasets	ja ¹
Löschen eines Datasets	ja
Hinzufügen einer Datei zu einem Dataset	nein ²
Entfernen einer Datei aus einem Dataset	ja
Erstellen eines Dataverses	ja ³
Bearbeiten eines Dataverses	ja
Löschen eines Dataverses	ja
Hinzufügen eines Benutzers zu einem Dataverse	ja
Entfernen eines Benutzers aus einem Dataverse	ja ¹
Erstellen einer Group	ja ¹
Bearbeiten einer Group	ja ¹
Erstellen eines Benutzers	ja
Entfernen eines Benutzers	ja ⁴

1: Es werden keine Details angezeigt.

2: Es wird nur ein Update ohne Details geloggt, kein Hinweis auf das Hinzufügen einer Datei.

3: In der "info"-Spalte steht weder der Name noch die ID des neuen Dataverses.

4: Die Funktion ist nur per API-Call möglich. Hierfür wird zwar ein API-Token verlangt, aber in der Log-Tabelle steht hinterher nicht, welcher Benutzer die Aktion ausgeführt hat.

Tabelle 6.15: Dataverse: Non-repudiation

Es werden zwar 13 der 14 Aktionen geloggt, aber in sieben der 13 gelogkten Aktionen fehlen wichtige Informationen. Insgesamt ist der Log etwas befriedigender als der CKAN-Log, aber dennoch nicht perfekt. Daraus folgt eine Bewertung von $2 = \textit{eher erfüllt}$.

Authenticity: Wie unter *Confidentiality* schon erwähnt, unterstützt Dataverse den Login mittels gängiger OAuth2-Provider wie Google, Github und Microsoft Azure AD sowie Shibboleth, welche wiederum den Einsatz von Zweifaktor-Authentifizierung erlauben, was für einen höheren Grad von Identitätssicherheit sorgt, als eine standardmäßige Authentifizierung per Benutzername und Passwort. OAuth2 ist der aktuelle Industriestandard für Authentifizierung, damit erhält Dataverse für dieses Kriterium eine Bewertung von $3 = erfüllt$.

Accountability: Wie auch beim Kriterium *Non-repudiation* ist die Bewertung dieses Kriteriums abhängig von der Qualität der Log-Datei. Im Log wird für jede geloggte Aktion festgehalten, welcher Benutzer sie ausgeführt hat. Da der Log aber z.T. nicht ausreichend genaue Informationen über die Veränderungen liefert, fällt die Bewertung mit $2 = eher erfüllt$ aus.

6.2.2.7 Maintainability (Wartbarkeit)

Modularity:

1. *Wurden Interfaces definiert und ordnungsgemäß verwendet?*

Mit Java wurde eine Programmiersprache gewählt, die aufgrund dessen, dass multiple Vererbung nicht erlaubt ist [16], bei komplexeren Projekten zur Vermeidung redundanten Codes auf die Verwendung von Interfaces angewiesen ist. Eine Überprüfung des Source Codes zeigt, dass zwar ca. 40 Interfaces definiert wurden, aber sich beim Benennen an keinerlei Konvention gehalten wurde. Darüber hinaus fiel in einer stichprobenartigen Überprüfung auf, dass der Source Code sehr sparsam kommentiert ist.

2. *Wird durch die Dokumentation und/oder Kommentare klar, welche Komponenten voneinander abhängen?*

Die Dokumentation enthält insgesamt nur ein einziges Diagramm. Dieses ist ein Sequenzdiagramm, das die beteiligten Komponenten des Systems in der Kommunikation mit einem anderen - optional in Dataverse einbindbaren - Projekt ("MakeDataCount") aufzeigen soll. Damit kann ein potenzieller Entwickler nicht viel anfangen, außer er möchte zufällig gerade an diesem Projekt arbeiten.

Insgesamt gibt es also keine Diagramme, die beim Verstehen des Codes helfen, nur dürftige Kommentare im Code und keine äußerlichen Hinweise bei Dateinamen, welche Dateien

zu Klassen und welche zu Interfaces gehören. Alles in allem also eine fast feindselige Umgebung für einen potenziellen Entwickler. Daraus folgt eine Bewertung von $0 = \textit{nicht erfüllt}$.

Reusability: Auch bei diesem System fällt die API großzügig aus, was die Wiederverwendung vieler Funktionen auf einfache Art und Weise ermöglicht. Leider wurden die Interfaces schlecht kommentiert und nicht äußerlich kenntlich gemacht, woraus sich eine Bewertung von $2 = \textit{eher erfüllt}$ ergibt.

Analysability: Wie schon erwähnt, gibt es überhaupt keine die Systemarchitektur beschreibenden Diagramme. Zudem ist der Code zumindest z.T. schlecht kommentiert und schlecht geordnet (es liegen ca. 150 .java-Dateien im Hauptverzeichnis ohne in Packages einsortiert zu sein). Das alles macht es sehr schwierig, das Projekt zu überblicken und zu analysieren. Aus alledem muss eine Bewertung von $0 = \textit{nicht erfüllt}$ folgen.

Modifiability: Das Modifizieren des Systems ist nicht vorgesehen. Stattdessen wird in der Dokumentation dazu geraten, externe Anwendungen mittels API auf die benötigten Daten zugreifen zu lassen und diese Anwendung dann mit einem “Explore”-Button einzubinden. Dabei wird explizit erwähnt, dass das Tool bei Verwendung des Explore-Buttons in einem neuen Tab und auf einer anderen Website aufgerufen wird. Dies führt zu einer Bewertung von $0 = \textit{nicht erfüllt}$.

6.2.2.8 Portability (Portierbarkeit)

Adaptability: Dataverse wurde im Rahmen der Arbeit für beide Anwendungsfälle auf unterschiedlichen Systemen mithilfe des offiziellen Dataverse-Docker-Images und der offiziellen Docker-Installationsanleitung für Dataverse in Docker-Containern installiert und gehostet. Wie in der Bewertung von CKAN schon erwähnt, ist Docker eine für Windows-, Mac- und Linux-Betriebssysteme erhältliche Virtualisierungssoftware. Daher sollte Dataverse für Docker auf jedem der drei Betriebssysteme funktionieren können. Im Rahmen der Arbeit wurde Dataverse auf virtuellen Maschinen mit unterschiedlichen Hardware-Spezifikationen (zwei oder vier Kerne, 4 oder 8GB RAM) betrieben und lief in beiden Versionen problemlos. Daraus folgt eine Bewertung von $3 = \textit{erfüllt}$.

Installability: Das System wurde nur in einer Version (4.19) auf einer Ubuntu-Version (18.04) und zwei verschiedenen virtuellen Maschinen installiert. Die Installation lief in

beiden Fällen reibungslos ab und sollte auch auf Windows und MacOS mit Docker funktionieren. Daraus folgt eine Bewertung von $3 = erfüllt$.

Replaceability: Wie bei einem Vergleich der beiden Tabellen 6.4 und 6.11 festzustellen ist, kann CKAN um eine Funktionalität nicht in vollem Umfang durch Dataverse ersetzt werden. Daraus folgt eine Bewertung von $2 = eher erfüllt$.

6.3 Vergleich der Evaluationsergebnisse und Abschluss

In der folgenden Tabelle finden sich die Ergebnisse der Evaluationen der beiden Kapitel 6.2.1 und 6.2.2 noch einmal zusammengefasst.

Kriterium	Punkte CKAN	Punkte Dataverse
1. Functional Suitability	2/3	2/3
1.1 <i>Functional Completeness</i>	2	2
2. Performance Efficiency	9/9	9/9
2.1 <i>Time Behaviour</i>	3	3
2.2 <i>Resource Utilization</i>	3	3
2.3 <i>Capacity</i>	3	3
3. Compatibility	6/6	6/6
3.1 <i>Co-existence</i>	3	3
3.2 <i>Interoperability</i>	3	3
4. Usability	16/18	13/18
4.1 <i>Appropriateness Recognizability</i>	2	2
4.2 <i>Learnability</i>	3	3
4.3 <i>Operability</i>	2	2
4.4 <i>User Error Protection</i>	3	1
4.5 <i>User Interface Aesthetics</i>	3	2
4.6 <i>Accessibility</i>	3	3
5. Reliability	9/9	6/9
5.1 <i>Maturity</i>	3	3
5.2 <i>Availabilty</i>	3	3
5.4 <i>Recoverability</i>	3	0
6. Security	7/15	12/15
6.1 <i>Confidentiality</i>	1	2
6.2 <i>Integrity</i>	3	3
6.3 <i>Non-repudiation</i>	1	2
6.4 <i>Authenticity</i>	1	3
6.5 <i>Accountability</i>	1	2
7. Maintainability	9/12	2/12
7.1 <i>Modularity</i>	2	0
7.2 <i>Reusability</i>	3	2
7.3 <i>Analysability</i>	1	0
7.4 <i>Modifiability</i>	3	0
8. Portability	9/9	8/9
8.1 <i>Adaptability</i>	3	3
8.2 <i>Installability</i>	3	3
8.3 <i>Replaceability</i>	3	2
gesamt	67/81	58/81

Tabelle 6.16: Vergleich der Evaluationsergebnisse

Wie in der Tabelle zu sehen ist, ist CKAN in sieben der acht Kategorien gleichwertig oder besser als Dataverse. Einzig in der Kategorie “Security” schneidet Dataverse besser ab, und zwar eindeutig.

Die in 3.1.1 angedeuteten Anforderungen der HAW an die Systeme werden besser von CKAN erfüllt als von Dataverse. Im Fall der HAW ist dies insbesondere die Anforderung der “gebündelten Darstellung von Daten und (schriftlichen) Ergebnissen”. Dass in CKAN Dateien online angesehen werden können, ohne sie herunterzuladen, ist dabei ein ausschlaggebender Faktor.

Auch die Anforderungen der HCHS werden besser von CKAN als von Dataverse erfüllt. Dataverse kommuniziert mit Fremdsystemen, was von vornherein schon ein Ausschlusskriterium ist. Zudem ist die für die HCHS nötige Erweiterung um einen Variablenkatalog nicht schön in die Anwendung zu integrieren, da weder die Dokumentation noch der Code besonders hilfreich bezüglich Modifizierungen ist und sogar dafür geworben wird, statt den Code zu verändern externe Anwendungen über einen Button einzubinden.

Die einzige richtige Schwachstelle von CKAN sind die Kriterien aus der Kategorie “Sicherheit”. Diese lassen sich aber recht simpel durch den Einsatz von bereits existierenden Extensions ausbessern. So gibt es bereits zwei verschiedene Extensions für OAuth2-Unterstützung [11]. Wenn dann noch die Seite mit der Benutzerübersicht so angepasst wird, dass sie nur für eingeloggte Sysadmins einsehbar ist, gibt es kaum noch Bedenken. Somit ist für beide Anwendungsfälle eindeutig CKAN zu empfehlen.

7 Zusammenfassung und Ausblick

Wie in 6.3 bereits erwähnt, ist CKAN für beide Anwendungsfälle das bessere System. In dieser Arbeit wurden Anforderungen zweier Organisationen formuliert und analysiert und dahingehende Defizite der zwei untersuchten Systeme identifiziert. Im nächsten Schritt wurde für eines der beiden Systeme eine Erweiterung konzipiert und implementiert, die das Defizit zumindest teilweise behoben hat. Dann wurde ein standardisiertes Bewertungsschema an die Bedürfnisse der Arbeit und die zur Verfügung stehenden Ressourcen angepasst und für beide Systeme angewandt und abschließend eine Empfehlung für beide Anwendungsfälle ausgesprochen.

Im Nachhinein waren einige der in der ISO 25010 beschriebenen Kriterien etwas zu aufwendig, um sie im Rahmen einer Bachelorarbeit auf eine wissenschaftlich valide Art und Weise zu überprüfen; eine Bewertung sollte beispielsweise hinsichtlich der Benutzbarkeit eines Systems nicht nur durch eine einzelne Person erfolgen, sondern mithilfe eines Usability-Tests inkl. Fragebogen mit mehreren Probanden durchgeführt werden. Dies war im Rahmen dieser Arbeit nicht möglich. Zudem hätte die Konzeption und Implementation der CKAN-Extension besser organisiert und strukturiert werden können.

Diese Arbeit hat gezeigt, dass mit den beiden untersuchten Systemen zwar auf einigermaßen sichere und reliable Weise Forschungsdaten gesammelt und geteilt werden können, aber Dataverse aufgrund schlechter Erweiterbarkeit sehr unflexibel in der Art der Nutzung ist und für CKAN eine stetigere Überprüfung und Anpassung der Dokumentation erforderlich ist, um das System produktiv nutzen zu können.

Da sich Dataverse nicht gut erweitern lässt und auch sonst in der Bewertung schlechter abgeschnitten hat als CKAN, ist es offenkundig, dass weitere experimentelle Arbeiten mit dem System sich nicht besonders anbieten. Zukünftige Arbeiten könnten - zumindest auf CKAN bezogen - zum Beispiel eine Einbindung der bestehenden Authentifizierungsmöglichkeiten der HAW oder eine "erzwungene" Einhaltung benutzerdefinierter Regeln für die Veröffentlichung von Daten und Arbeiten mithilfe einer angepassten Geschäftslogik behandeln.

Literaturverzeichnis

- [1] *Adding CSS and JavaScript files using Fanstatic - CKAN 2.8.4 documentation.* – URL <https://docs.ckan.org/en/2.8/theming/fanstatic.html>. – letzter Zugriff: 13.07.2020
- [2] *Apache Solr Market Share and Competitor Report | Compare to Apache Solr, Apache Lucene, Swifttype.* – URL <https://www.datanyze.com/market-share/enterprise-search--287/apache-solr-market-share>. – letzter Zugriff: 23.06.2020
- [3] *Best Server And Application Response Time Monitoring Tools + Guide - DNSStuff.* – URL <https://www.dnsstuff.com/response-time-monitoring>. – letzter Zugriff: 30.04.2020
- [4] *BSI für Bürger - Zwei-Faktor-Authentisierung.* – URL https://www.bsi-fuer-buerger.de/BSIFB/DE/DigitaleGesellschaft/OnlineBanking/Zwei_Faktor_Authentisierung/Zwei-Faktor-Authentisierung_node.html. – letzter Zugriff: 23.06.2020
- [5] *ckan - the open source data portal software.* – URL <https://ckan.org>. – letzter Zugriff: 10.04.2020
- [6] *CVE security vulnerability database.* – URL <https://www.cvedetails.com>. – letzter Zugriff: 06.06.2020
- [7] *data.gov.au - beta.* – URL <https://data.gov.au>. – letzter Zugriff: 10.04.2020
- [8] *The Dataverse Project.* – URL <https://dataverse.org>. – letzter Zugriff: 17.04.2020
- [9] *Datenmanagement - Wikipedia.* – URL <https://de.wikipedia.org/wiki/Datenmanagement>. – letzter Zugriff: 07.04.2020

- [10] *Directory Traversal* - *Wikipedia*. – URL https://de.wikipedia.org/wiki/Directory_Traversal. – letzter Zugriff: 20.06.2020
- [11] *Find CKAN Extensions* - *CKAN Extensions*. – URL <https://extensions.ckan.org/>. – letzter Zugriff: 23.06.2020
- [12] : *GitHub* - *okfn/csv.js: Simple, ultra-light (10kb) JS library for CSV parsing [...]*. – URL <https://github.com/okfn/csv.js/>
- [13] *Hamburg City Health Study*. – URL <https://hchs.hamburg>. – letzter Zugriff: 29.03.2020
- [14] *Home* - *Recline Data Explorer and Library*. – URL <https://okfnlabs.org/recline/>. – letzter Zugriff: 23.05.2020
- [15] *ISO 25000 PORTAL*. – URL <https://iso25000.com/>. – letzter Zugriff: 10.04.2020
- [16] *Java (Programmiersprache)* - *Wikipedia*. – URL [https://de.wikipedia.org/wiki/Java_\(Programmiersprache\)](https://de.wikipedia.org/wiki/Java_(Programmiersprache)). – letzter Zugriff: 23.06.2020
- [17] *Joint Information Systems Committee* - *Wikipedia*. – URL https://de.wikipedia.org/wiki/Joint_Information_Systems_Committee. – letzter Zugriff: 10.04.2020
- [18] : *Library* - *Home* - *Recline Data Explorer and Library*. – URL <https://okfnlabs.org/recline/docs/>
- [19] *Likert-Skala* - *Wikipedia*. – URL <https://de.wikipedia.org/Likert-Skala>. – letzter Zugriff: 01.05.2020
- [20] : *Loading data from different sources using Backends* - *Tutorial* - *Recline Data Explorer and Library*. – URL <https://okfnlabs.org/recline/docs/tutorial-backends.html>
- [21] *Managing Research Data Programme 2011-13*. – URL https://webarchive.nationalarchives.gov.uk/20140703054830/http://www.jisc.ac.uk/whatwedo/programmes/di_researchmanagement/managingresearchdata. – letzter Zugriff: 10.04.2020
- [22] *Overview* - *CKAN 2.8.2 documentation*. – URL <https://docs.ckan.org/en/2.8/>. – letzter Zugriff: 15.06.2020

- [23] *Python (Programmiersprache)* - *Wikipedia*. – URL [https://de.wikipedia.org/wiki/Python_\(Programmiersprache\)](https://de.wikipedia.org/wiki/Python_(Programmiersprache)). – letzter Zugriff: 23.06.2020
- [24] : *recline/demos/multiview at master - mattfullerton/recline - GitHub*. – URL <https://github.com/mattfullerton/recline/tree/master/demos/multiview>
- [25] : *Resources - CKAN 2.8.4 documentation*. – URL <https://docs.ckan.org/en/2.8/contributing/frontend/resources.html>
- [26] *styleguide - Style guides for Google-originated open-source projects*. – URL <https://google.github.io/styleguide/pyguide.html>. – letzter Zugriff: 15.06.2020
- [27] : *Tabulator*. – URL <http://tabulator.info/docs/4.7/data#array>
- [28] AHMED, T. M. ; BEZEMER, C. ; CHEN, T. ; HASSAN, A. E. ; SHANG, W.: Studying the Effectiveness of Application Performance Management (APM) Tools for Detecting Performance Regressions for Web Applications: An Experience Report. In: *2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR)*, 2016, S. 1–12
- [29] AUBIN, Sophie: *Dataverse*. 2018. – URL https://www.circasa-project.eu/content/download/3839/37420/version/1/file/20190207_CIRCASA_first%20General%20Assembly_3-2A_Jean-Fran%20%20Soussana_WP1%20DataVerse.pdf. – Vortrag im Rahmen des CIRCASA-Projekts
- [30] CROSAS, Mercè: *The Dataverse Network®: An Open-Source Application for Sharing, Discovering and Preserving Data*, 2011
- [31] GIBBS, Eric ; LIN, Lin ; QUIGLEY, Elizabeth ; TANG, Rong: *Dataverse Usability Evaluation: Final Report*. (2013), 05/29, S. 1–18
- [32] HOLZ AUF DER HEIDE, Bernd: Welche software-ergonomischen Evaluationsverfahren können was leisten? In: RÖDIGER, Karl-Heinz (Hrsg.): *Software-Ergonomie '93: Von der Benutzungsoberfläche zur Arbeitsgestaltung*. Stuttgart : B.G.Teubner, 1993, S. 157–171
- [33] MIGUEL, Jose ; MAURICIO, David ; RODRIGUEZ, Glen: A Review of Software Quality Models for the Evaluation of Software Products. In: *International journal of Software Engineering & Applications* 5 (2014), 11, S. 31–54

- [34] SANTUCCI, Guisepppe: *Software Quality (and ISO 25010) Part II.* – URL <https://www.scribd.com/document/385648250/08-Quality-25010-II-pdf>. – letzter Zugriff: 01.05.2020
- [35] WINN, Joss: *Open data and the academy: an evaluation of CKAN for research data management*, 2013

Erklärung zur selbstständigen Bearbeitung einer Abschlussarbeit

Gemäß der Allgemeinen Prüfungs- und Studienordnung ist zusammen mit der Abschlussarbeit eine schriftliche Erklärung abzugeben, in der der Studierende bestätigt, dass die Abschlussarbeit „— bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit [(§ 18 Abs. 1 APSO-TI-BM bzw. § 21 Abs. 1 APSO-INGI)] — ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt wurden. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich zu machen.“

Quelle: § 16 Abs. 5 APSO-TI-BM bzw. § 15 Abs. 6 APSO-INGI

Erklärung zur selbstständigen Bearbeitung der Arbeit

Hiermit versichere ich,

Name: _____

Vorname: _____

dass ich die vorliegende Bachelorarbeit – bzw. bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit – mit dem Thema:

Vergleichende Evaluation von Open-Source Software für Forschungsdatenmanagement

ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort

Datum

Unterschrift im Original