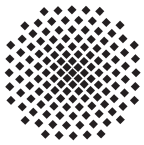

H L R S

Institut für
Höchstleistungsrechnen

FORSCHUNGS- UND ENTWICKLUNGSBERICHTE

FORSCHUNGSDATENMANAGEMENT IM KONTEXT DUNKLER DATEN IN DEN SIMULATIONSWISSENSCHAFTEN

Björn Schembera

ISSN 0941 - 4665 Dezember 2019 HLRS-21

Höchstleistungsrechenzentrum
Universität Stuttgart
Prof. Dr.-Ing. Dr. h.c. Dr. h.c. Prof. E.h. Michael M. Resch
Nobelstrasse 19 - 70569 Stuttgart
Institut für Höchstleistungsrechnen

FORSCHUNGSDATENMANAGEMENT IM KONTEXT DUNKLER DATEN IN DEN SIMULATIONSWISSENSCHAFTEN

von der Fakultät Energie-, Verfahrens- und Biotechnik der Universität Stuttgart zur Erlangung der Würde eines Doktor-Ingenieurs (Dr.-Ing.) genehmigte Abhandlung

vorgelegt von

Björn Schembera
aus Rosenheim

Hauptberichter:	Prof. Dr.-Ing. Dr. h.c. Dr. h.c. Prof. E.h. Michael M. Resch
Mitberichter:	Prof. Dr. Thomas Ludwig
Tag der Einreichung:	19. Juni 2019
Tag der mündlichen Prüfung:	20. Dezember 2019

D93

ISSN 0941 - 4665 Dezember 2019 HLRS-21

Denn die einen sind im Dunkeln
Und die andern sind im Licht.
Und man siehet die im Lichte
Die im Dunkeln sieht man nicht.

Die Moritat von Mackie Messer, Bertolt Brecht

Danksagung

Unterstützung fachlicher, kollegialer, freundschaftlicher oder familiärer Natur war in der Zeit meines Promotionsvorhabens unerlässlich. Den dahinter stehenden Personen möchte ich an dieser Stelle gebührend danken.

Meinem Doktorvater *Prof. Dr. Michael M. Resch* danke ich herzlich für die Motivation und die Förderung meiner Ideen. Seine Art, sich von fachlichen Grenzen nicht im Denken hemmen zu lassen, war für mein multidisziplinäres Promotionsvorhaben äußerst bereichernd. Für die Übernahme der Mitberichterschaft möchte ich *Prof. Dr. Thomas Ludwig* danken und insbesondere auch für die Einladung an das DKRZ im Februar 2019, um dort den Stand meiner Arbeit zu diskutieren.

Besonders möchte ich *Dr. Thomas Bönisch*, der mich in meinem Promotionsvorhaben auch fachlich unterstützt hat, sowie *Manuela Wossough* für die Bereitstellungen einiger statistischer Daten, dem sorgfältigen Korrekturlesen sowie für die Unterstützung bei meiner Verteidigungsfeier meinen herzlichen Dank aussprechen. Meinem ehemaligen Bürokollegen *Dr. Xuan Wang* danke ich für die gegenseitige Motivation und den Austausch zum jeweiligen Dissertationsvorhaben, sowie *Oleksandr Shcherbakov* für hilfreiche Hinweise bzgl. der Cray Urika. *Dr. Juan Durán* möchte ich für die gemeinsame Arbeit zum Thema Dunkler Daten danken, die als Ausgangspunkt und Rahmen für diese Dissertation diente. Weiterhin bedanke ich mich bei *Dr. Andreas Kaminski*, *Björn Dick* und *Freddie Klank*.

Ein besonderer Dank geht an *Dr. Dorothea Iglezakis* für die gute Zusammenarbeit im äußerst produktiven Projekt *DIPL-ING*, aus dem viele Denkanstöße stammen, die schließlich in meiner Dissertation umgesetzt wurden, sowie für das inhaltliche Korrekturlesen. In diesem Zuge möchte ich *Björn Selent*, *Dr. Gernot Bauer* und *Hamzeh Kraus* speziell für die Bereitstellung der Testfälle aus den Fachdispziplinen dankend erwähnen. *Dr. Martin Horsch* spreche ich meinen herzlichen Dank für das sorgfältige inhaltliche Korrekturlesen aus.

Aus meinem Freundeskreis möchte ich mich herzlich für das sorgfältige und zügige Korrekturlesen verschiedener Stände und Teile bei *Dr. Kaja Tulatz*, *Lisa Neher*, *Marcus Brennenstuhl*, *Michel Saalbach* sowie *Dr. Simone Heinold* bedanken. Bei *Linda Weinmann* bedanke ich mich herzlich für die Hilfe bei der Erstellung der Grafiken. Weiterhin gebührt mein Dank meinen Freundinnen und Freunden *Simon Erne*, *Jan-Dominik Kohn*, *Thorsten Mertn*, *Holger Mertn*, *Florian Haspel*, *Dr. Anja Kuhn*, *Prof. Dr. Jan Paul Lindner*, *Katrin Saalbach*, *Dr. Matthias Ölschläger*, *Tanja Kotik* und *Raimund Huber* für die mentale Unterstützung und Zerstreuung während der Dissertation, aus der ich beständig Kraft schöpfen konnte. Ohne meine Freundinnen und Freunde – gleich ob durch deren direkte oder indirekte Unterstützung – wäre das Dissertationsvorhaben ein unmögliches Unterfangen gewesen.

Der *Rockfabrik Ludwigsburg*, die zum 1. Januar 2020 nach 36 Jahren Betrieb leider gezwungenermaßen schließen musste, bzw. ihrem Team und stellvertretend ihrem

Kopf *Johannes Rossbacher*, danke ich von ganzem Metal-Herzen für die Zeiten während Studium und Promotion, die ich dort mit Muße und (lauter) Musik verbringen durfte.

Schließlich gebührt mein besonderer Dank meinen Eltern *Dieter* und *Elisabeth*. Sie waren es, die mir 1997 mein ersters Programmierbuch schenkten und mich während Studium und Promotion stets unterstützten, mir Vertrauen schenkten und mich dabei meinen eigenen Weg gehen ließen. Ihre Unterstützung ist für mich immer unersetzbar gewesen.

Stuttgart, im Juni 2020 Björn Schembera

Zusammenfassung

Forschungsdatenmanagement im Höchstleistungsrechnen und speziell in den Ingenieurwissenschaften steht vor einigen Herausforderungen: Die typischerweise großen Datenmengen, fehlende Metadatenmodelle zur hinreichenden Beschreibung der Daten sowie mangelnde Integration in den Forschungsprozess, die durch das gänzliche Fehlen von Anreizen zur Datendokumentation noch verstärkt werden. Bedingt durch diese Herausforderungen entstehen im gegenwärtigen Arbeitsprozess zwangsläufig *dunkle Daten*. Dies sind Daten, die nicht mit Metadaten versehen oder technisch nicht mehr zugänglich sind und somit nicht mehr universell verstehbar, verfügbar oder nutzbar sind. Sie sind dem heutigen Datenlebenszyklus durch o.g. Missmanagement inhärent.

In der vorliegenden Dissertation wurde der bereits bestehende Begriff der *dunklen Daten* auf den Bereich des Höchstleistungsrechnens (HPC) erweitert. Dunkle Daten wurden in Speichersystemen im HPC in ihrer zeitlichen Entwicklung nachgewiesen und mit Statistiken belegt. Dabei haben dunkle Daten einige Implikationen: Neben rechtlichen und technischen Problemen stehen Probleme der Verantwortung. Verantwortungsprobleme treten z.B. dort auf, wo nicht klar geregelt ist, was mit den Daten nach Projektende geschieht. Darüber hinaus blockieren dunkle Daten Ressourcen. Sie stehen dem Verständnis von FAIRen (*F*indable/findbaren, *A*ccessible/zugreifbaren, *I*nteroperable/interoperablen und *R*e-usable/nachnutzbaren) Daten diametral gegenüber. Ihr Auftreten ist ein Zeichen dafür, dass der Datenlebenszyklus insofern verkürzt ist, als dass das Datenmanagement nach Ende des Rechenprojektes endet und die Daten nicht zur Nachnutzung im Sinne der guten wissenschaftlichen Praxis bereitgestellt werden können.

In der Dissertation werden Konzepte erarbeitet, um dunkle Daten zu verringern bzw. zu vermeiden. Diese wurden anhand der Anwendungsfälle der Thermo- und Aerodynamik entwickelt. Dazu wurde das Metadatenmodell *EngMeta* entwickelt, welches eine Datendokumentation in den computergestützten Ingenieurwissenschaften ermöglicht. Dabei werden, neben den eigentlichen Forschungsdaten, der Forschungsprozess selbst sowie einige Elemente der Modellbildung beschrieben. Dies bedeutet, dass zusätzlich zu technischen und deskriptiven auch fachspezifische und Prozessmetadaten abgebildet werden. Das Metadatenmodell wurde anhand der zwei Anwendungsfälle der thermodynamischen und aerodynamischen Simulation von einem Objektmodell ausgehend entwickelt und als XML-Schema implementiert, um strenge Validierbarkeit und Maschinenlesbarkeit zu garantieren.

Darauf aufbauend wurde eine *automatisierte Metadatenerfassung* entworfen, welche die schon durch die Simulationscodes vorhandenen strukturiert oder teilstrukturiert vorliegenden Metadaten direkt nach Erzeugung ermittelt und in das Metadatenmodell *EngMeta* überführt. Dazu musste zunächst die automatisierte Erfassbarkeit der verschiedenen Metadatenkategorien geklärt werden. Technische Metadaten und Prozessmetadaten lassen sich besonders gut erfassen. Fachspezifische Metadaten sind teilweise und deskriptive Metadaten nur sehr schlecht automatisiert erfassbar. Die Erfassung wurde so implementiert, dass sie sich einfach in den wissenschaftlichen Arbeitsprozess integrieren lässt, indem sie direkt nach dem Simulationslauf ausführbar ist. Sie ist insofern generisch, als sie mittels einer Konfigurationsdatei arbeitet. So ist keinerlei Entwicklungsarbeit zu leisten, sollen die Metadaten neuer Simulationscodes erfasst werden. Damit soll die Hemmschwelle der Nutzung möglichst gering gehalten werden.

Da Metadatenprodukte wie ein Metadatenmodell ohne die entsprechenden Prozesse nutzlos sind, wurde die Rolle des *Scientific Data Officers* eingeführt und auf die entwickelten Konzepte bezogen. Diese Rolle kann auf allen Ebenen einer Organisationseinheit wirken, also z.B. auf Gruppenebene ebenso wie auf Universitätsebene. Diese Rolle hat dabei die Verantwortung über das Forschungsdatenmanagement innerhalb einer Organisationseinheit. Auf Seiten der technischen Verantwortung soll sie Metadatendokumentation befördern, periodisch das Dateninventar auf dunkle Daten prüfen sowie lokalisierte dunkle Daten in ihre Obhut nehmen. Im Gegensatz zur Position des Systemadministrators ist ihr Charakteristikum die Sichtbarkeit innerhalb der Institution, da sie für alle Belange des Forschungsdatenmanagements verfügbar sein muss und inbesondere zur Sichtbarmachung von dunklen Daten dient. Dabei orientiert sie sich an den in der Dissertation aufgestellten *Entscheidungskriterien*, welche Daten wie lange gespeichert werden sollten. Diese finden sich für die Fälle ausgearbeitet, in denen aus den Daten eine Publikation folgte, für Daten ohne Publikation und für dunkle Daten.

Die Ansätze wurden einer Evaluation unterzogen. Es zeigte sich, dass das Metadatenmodell *EngMeta* qualitative Kriterien für ein gutes Metadatenmodell erfüllt und nicht auf die Anwendungsbereiche Thermo- und Aerodynamik beschränkt ist. Dies wird einerseits durch die Integration von Standardmetadatenmodellen wie DataCite oder PREMIS erreicht und liegt andererseits an der Konzeption: Die fachspezifischen Metadatenelemente, die kontrollierte und gemessene Variablen beschreiben, sowie die Rechnerumgebung und die Methodik finden sich in allen ingenieurwissenschaftlichen Disziplinen, die Computersimulation nutzen. Die *automatisierte Metadatenerfassung* wurde auf Systemen getestet, die auch für Simulation genutzt werden, wie z.B. der Cray XC40 Hazel Hen. Darüber hinaus wurde die automatisierte Metadatenerfassung für die Dateiformate EAS3, NetCDF und von GROMACS, getestet. Schließlich wurde die Integration in den Forschungsprozess evaluiert, indem die Erfassung nach dem Simulationslauf automatisch ausgeführt wurde und die erfassten Metadaten gemeinsam mit den Daten automatisiert in ein Repositorium gespielt wurden.

Abstract

Research data management in high-performance computing, particularly in computational engineering, faces several challenges. The typically large quantities of data involved, an outstanding need for metadata models capable of describing the data sufficiently and a lack of integration into the scientific workflows. These problems are intensified by the complete absence of incentives for data documentation. As a result of these challenges, an inevitable product of today's working processes is dark data; i.e. data that is either not annotated with metadata or that is technically inaccessible. Hence, this data is no longer universally understandable, accessible or useful. Because of the aforementioned mismanagement of the current data life cycle, dark data have become an inherent byproduct of the research process.

In this doctoral thesis, the already existing concept of dark data is extended to the field of high-performance computing (HPC). It identifies the development of dark data in HPC storage systems across time and proves its existence using statistical methods. The existence of dark data has several implications: In addition to legal issues and technical problems, it raises additional problems of accountability, which arise for example when data are not clearly managed after concluding a project. Moreover, dark data can freeze resources. They exist in fundamental opposition to the paradigm of FAIR (findable, accessible, interoperable, re-usable) data. Their occurance is a sign of a incomplete data life cycle in which research data management ends concurrently with the end of a project and data is not made available for reuse in accordance with scientific best practices.

The dissertation presents concepts for minimizing and avoiding the production of dark data. The metadata model *EngMeta* has been developed to enable data documentation for computational engineering applications. In addition to the research data itself, the research process and elements involved in building the model are described. Developing *EngMeta* can only be accomplished when, in addition to technical and general descriptive metadata, process metadata as well as domain-specfic metadata is used. The metadata model was developed in the context of two use cases based in thermodynamic simulation and aerodynamic simulation. Based on an object model, the description was implemented as an XML schema, guaranteeing strict validation and machine-readability.

Building on this metadata model, an *automated metadata extraction* was designed and implemented. In this approach, structured or semi-structured metadata that is available from the simulation output files is parsed immediately following the simulation and transfered to *EngMeta*. This requires first the determining of the

different metadata categories. Technical and process metadata are easy to extract. Using automated methods, domain-specific metadata is partially extractable, while descriptive metadata is typically not extractable. The extraction was implemented in a way that makes it easy to integrate into a scientific workflow, as it can be configured to automatically run after the simulation run is complete. The extraction is generic in the sense that it works using a configuration file. This means that no development work is necessary to extract metadata associated with new simulation codes or other calculations. This should lower the barriers to usage.

Becauase metadata products such as metadata models are useless without corresponding processes, the role of the *Scientific Data Officer* is introduced with respect to the developed concepts. This role can operate at all levels of an organisation; e.g. on the team level or at a university-wide level. This role has the responsibility for research data management within an organizational unit. Its technical responsibilities include promoting metadata annotation, periodically checking the data inventory for dark data, taking charge of localized dark data, and organizing education and training activities. Unlike system administrators, one key characteristic of this position is its visibility, as it must be available to address all concerns related to research data management and has the explicit aim of making dark data visible. To accomplish this, the position applies the *decision criteria* that have been developed within this dissertation which provide guidance on which data should be stored and for how long. These criteria address cases in which data form the basis of a publication, data are collected without being related to a publication, or constitute dark data.

The approaches described here have undergone an evaluation. It was shown that the metadata model *EngMeta* fulfills the criteria of a good metadata model and that its applicability is not limited to thermo- and aerodynamics research. On the one hand, this was achieved by integrating metadata standards such as DataCite or PREMIS. On the other, this is the result of the way in which it was conceived: The domain-specific metadata elements, which describe controlled and measured variables, as well as the description of the simulation process with respect to the computational environment and the methodology is not bound to these two fields but can be found in all disciplines that use computer simulation. Automated metadata extraction was tested on systems that are also used for actual simulations; e.g. on the Cray XC40 Hazel Hen or on the bwUniCluster. Moreover, the automated metadata extraction was evaluated for simulation codes with respect to the data formats GROMACS, EAS3, NetCDF. Finally, integration into the scientific workflow was also evaluated. It was shown that the extraction could be executed immediately following the simulation itself, and the extracted metadata could be automatically transferred together with the data to a repository.

Inhaltsverzeichnis

Kapitel 1

Einleitung und Motivation

1.1 Einleitung

Ein Teil des heutigen wissenschaftlichen Arbeits- und Erkenntnisprozesses basiert wesentlich auf datenerzeugender Simulation (Gray u. a., 2005; Hey und Trefethen, 2003; Hey, Tansley und Tolle, 2009), welche mittlerweile als dritte Säule der wissenschaftlichen Methodik zwischen Theorie und Experiment eingeordnet wird (Winsberg, 2003; Winsberg, 2009). Datenproduktion und -auswertung mittels Computermodellen auf Höchstleistungsrechnern ist für sie typisch. Von einer Startkonfiguration ausgehend wird das Verhalten eines Systems, welches durch ein Modell beschrieben ist, hergestellt und beobachtet. Diese Beobachtung findet durch das Herausschreiben von Systemeigenschaften auf persistentem Speicher wie Festplatten zu bestimmten Zeitpunkten statt. Dabei nehmen die so produzierten Daten, im Folgenden *Forschungsdaten* oder nur *Daten* genannt, die zentrale Rolle als Grundlage und Ausgangspunkt aller wissenschaftlicher Erkenntnis in diesem Bereich ein.

In der Molekulardynamik können beispielsweise Trajektorien von Molekülen basierend auf Computermodellen berechnet werden (Bungartz u. a., 2013). Abbildung 1.1 zeigt die Visualisierung einer Nanoröhre, die mittels eines Computermodells berechnet wurde. Nanoröhren können z.B. in der Medizin eingesetzt werden, um Wirkstoffe im Blut an einen gewissen Ort zu bringen. Durch Simulation können neue Fälle durchgespielt, kritische Situationen erkannt und neue Erkenntnisse gewonnen werden, indem Parameter, wie z.B. die Temperatur, der Druck oder die Bindungswinkel der Moleküle variiert werden. Für alle durchgespielten Fälle werden neue Daten erzeugt und ausgewertet. Allerdings spiegelt sich insbesondere im Höchstleistungsrechnen (HPC) die Wichtigkeit von Forschungsdaten nicht im Management dieser Forschungsdaten wider. Während der Datenproduktions- und auswertungsphase muss noch eine Datenverwaltung etabliert sein, die sich jedoch meistens auf eine individuelle, im besten Fall institutsweit einheitliche Beschreibung der Daten durch Datei- und Verzeichnisnamen beschränkt. Nach dieser Phase ist das Interesse an Datenmanagement allerdings gering. Es fehlen Anreize, finanzielle Mittel und technische Möglichkeiten, die Daten FAIR, d.h. findbar (**F**indable), zugreifbar (**A**ccessible), interoperabel (**I**nteroperable) und nachnutzbar (**R**e-usable) (Wilkinson u. a., 2016) zu halten. Zwar schreiben Richtlinien, wie z.B. die Empfehlung Nr. 7

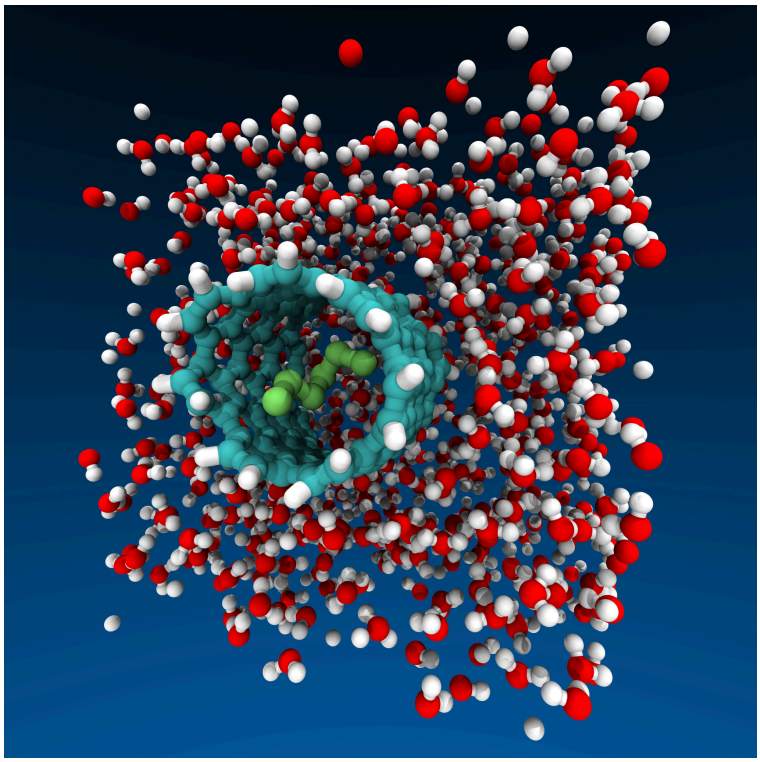

Abbildung 1.1: Visualisierung einer Nanoröhre. (Gernot Bauer/ITT)

der DFG zur Sicherung guter wissenschaftlicher Praxis vor, dass Primärdaten „auf haltbaren und gesicherten Trägern in der Institution, wo sie entstanden sind, zehn Jahre lang aufbewahrt werden“ (DFG, 2013, S. 21) sollen. Dies geht jedoch oft an der Realität vorbei, weil dem technische und organisatorische Herausforderungen, die noch nicht gelöst sind, entgegenstehen. Hierzu zählen z.B. die im HPC typischen großen Datenmengen im Bereich von Tera- oder Petabytes, welche die langfristige Speicherung finanziell und technisch aufwendig machen. Ebenso gibt es im Bereich HPC kein einfach nutzbares Repositorium, das erweiterte Datenverwaltungsmechanismen zur Verfügung stellt[1]. Darüber hinaus fehlen Metadatenmodelle, die auf HPC und eine ihrer Hauptdisziplin – das Ingenieurwesen – zugeschnitten sind. All diese Herausforderungen brechen sich Bahn in *dunklen Daten*, welche somit pars pro toto für die Probleme bzgl. der Forschungsdaten im HPC stehen. Dunkle Daten sind Daten, die sich der Sichtbarkeit und dem Zugriff entziehen – also Daten, die einerseits unterindiziert sind, d.h. nicht mit Metadaten zur Beschreibung angereichert sind, und/oder solche, die durch Inaktivität verloren gehen, indem z.B. eine Doktorandin oder ein Doktorand das Institut verlässt, ohne die von ihr erzeugten Daten einer anderen Person zu überantworten. Der allgemeine Begriff der dunklen Daten wurde in (Heidorn, 2008) eingeführt und wird im Rahmen der vorliegenden Arbeit für den Bereich des HPC diskutiert und erweitert. Dabei wurden dunkle Daten in den Speichersystemen des Höchstleistungsrechenzentrums (HLRS) nachgewiesen. Anforderungen an Forschungsdatenmanagement (FDM) müssen folglich so gestaltet

[1] `https://www.re3data.org/`, Zugriff 27. Juni 2018.

sein, dass dunkle Daten verhindert und Datenmanagement FAIR gestaltet werden kann. Es werden Lösungen einerseits technischer, andererseits organisatorischer Art vorgestellt, welche im Rahmen der Dissertation entwickelt wurden. Auf der Seite der technischen Lösungsansätze wurde *EngMeta*, ein Metadatenmodell für die Ingenieurwissenschaften zur Beschreibung der Forschungsdaten entwickelt sowie eine automatisierte Metadatenerfassung nah am Forschungsprozess implementiert. Das Metadatenmodell wurde in ein mehrschichtiges Datenmanagementsystem an der Universität Stuttgart integriert. Dadurch ist es möglich, die Daten direkt nach Generierung mit Metadaten zu versehen, was dunkle Daten durch Unterindizierung verhindert. Auf organisatorischer Seite steht der Scientific Data Officer, welcher als spezifischer Datenkurator im HPC wirkt und dunkle Daten, die durch Inaktivität entstehen, verhindern soll. Diese neue Rolle wird ergänzt durch Entscheidungskriterien, wie lange Daten aufbewahrt werden sollten und wann bzw. nach welcher Prozedur dunkle Daten eventuell gelöscht werden können.

1.2 Forschungsdaten im Kontext von HPC

1.2.1 Simulation

Da Forschungsdaten in diesem Kontext das direkte Produkt von Simulation sind und die Konzepte in der Dissertation am Forschungsprozess selbst ansetzen, wird Simulation hier begrifflich definiert. Simulation ist, gemäß Hartmann, die „Imitation eines Prozesses durch einen anderen Prozess. In dieser Definition bezieht sich der Begriff *Prozess* lediglich auf ein Objekt oder System, dessen Zustand sich über die Zeit ändert“ (eigene Übersetzung) (Hartmann, 1996, S. 83). Simulation ist somit ein System, welches das dynamische Verhalten eines anderen Systems imitiert. Sie kann auch als Reproduktion des Verhaltens eines Realweltphänomens oder -systems über die Zeit gesehen werden. Diese Imitation bzw. Reproduktion benötigt ein *Modell*, welches einige wesentliche Charakteristika und Funktionen des Phänomens oder Systems repräsentiert. Das *System* oder auch *Zielsystem* ist das eigentliche Forschungsobjekt, das analysiert werden soll. Das Modell ist eine abstrakte Abbildung dieses Zielsystems, das eine gewisse Überlappung mit dem Zielsystem hat. Dabei ahmt es gewisse Eigenschaften, Prozesse und Funktionen des Zielsystems nach. Dieser Zusammenhang ist der Kern von Modellbildung und Simulation und ist in Abbildung 1.2 dargestellt. Das Zielsystem wird auf ein Modell abgebildet, welches mittels Repräsentation einige Charakteristika imitiert. Dieses Modell wird implementiert und zur Ausführung gebracht, wobei Daten produziert werden. Dies ist neben dem Experiment ein möglicher Zugang, um Phänomene und Systeme in der realen Welt zu verstehen. Der Simulationsbegriff ist in der von Hartmann vorgeschlagenen Definition nicht auf Computersimulation beschränkt, wird aber im Folgenden nur in diesem Sinne verwendet.

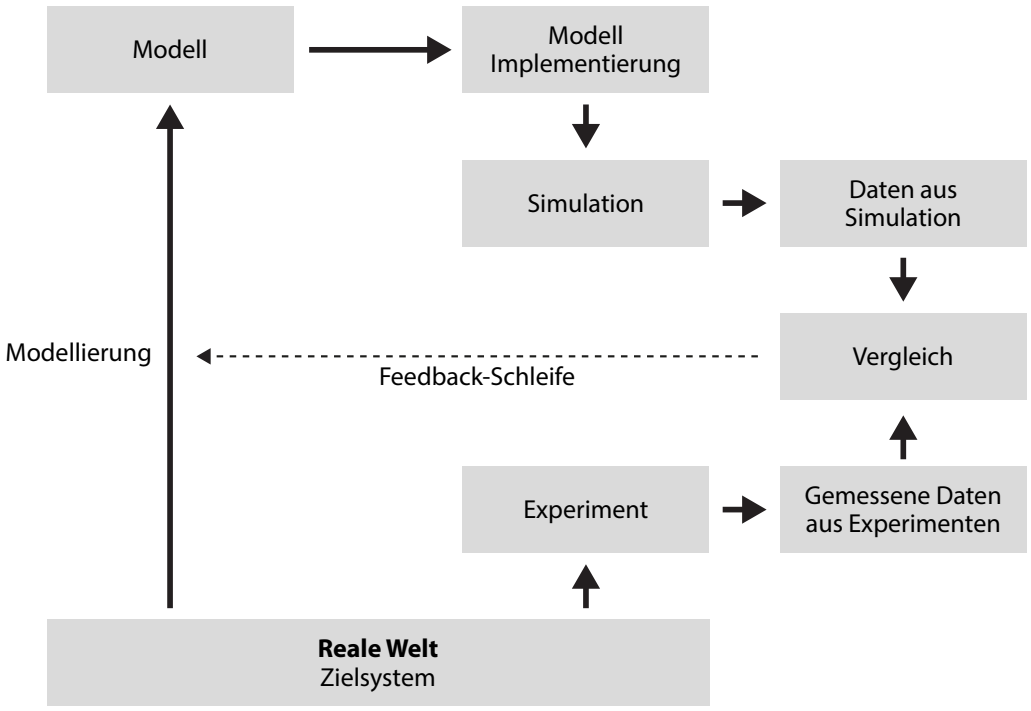

Abbildung 1.2: Der Modellierungs- bzw. Simulationsprozess: Abbildung eines Zielsystems aus der realen Welt auf ein Modell, welches implementiert und simuliert wird. Dabei werden Daten erzeugt. Erweiterung der Darstellung nach (Hasse und Lenhard, 2017).

1.2.2 Der HPC-Arbeitsprozess

Wie im vorigen Abschnitt aufgezeigt wurde, sind die abstrakten Ausgaben bzw. Ergebnisse von Simulation in Form von Daten vorhanden. Simulation kann in diesem Sinne auch als Datenproduktion gesehen werden. Das Ziel der wissenschaftlichen Arbeit im HPC ist der Erkenntnisgewinn mittels Auswertung dieser Daten, die in einer Publikation mit den zusammengefassten kommunizierbaren Resultaten kulminiert. Die Phasen der Arbeit mit den jeweils relevanten Ein- und Ausgabedaten sind in Abbildung 1.3 dargestellt. In der Hauptphase werden die Rohdaten und Restart-Dateien[2] von einer Startkonfiguration ausgehend durch den Simulationscode und die Steuerskripten erzeugt. Die Rohdaten werden in einem zweiten Schritt nachbearbeitet und z.B. durch Auswertungsskripte oder weitere Programme nach interessanten Eigenschaften durchsucht. Sie werden so zu gefilterten oder ausgewerteten Daten. Diese gefilterten Daten werden schließlich durch weitere Programme und Skripten analysiert, was zu aggregierten Daten, Tabellen oder Plots führt. Solche Darstellungen werden schließlich im Rahmen einer Publikation veröffentlicht, in der die Ergebnisse der Analyse kommuniziert und somit Aussagen über das simulierte Zielsystem getroffen werden.

[2] Restart-Dateien entsprechen Zwischenständen der Simulation, die zu einem späteren Zeitpunkt wieder aufgenommen werden können, falls die Simulation unerwartet abbricht.

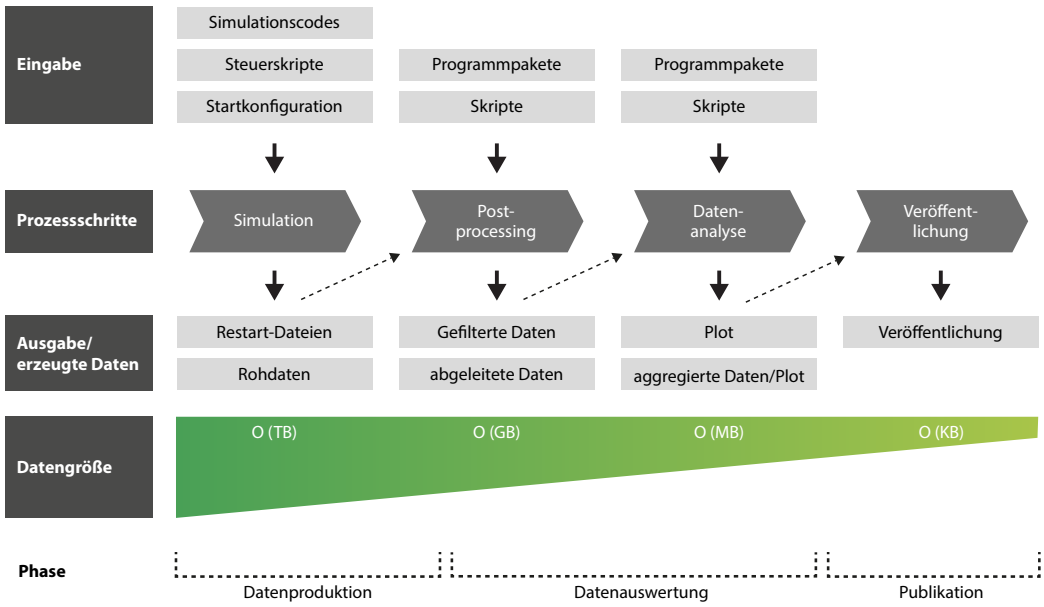

Abbildung 1.3: Die Phasen des wissenschaftlichen Prozesses im HPC gliedern sich in *Datenproduktion*, *Datenauswertung* und *Publikation*. Es werden jeweils entsprechende Ausgabedaten in der entsprechenden Größenordnung erzeugt.

In Abbildung 1.3 ist auch das Datenvolumen in der jeweiligen Phase dargestellt. In der ersten Phase der Datenproduktion werden dabei Daten aus dem Modell heraus oder von wenigen Startdaten ausgehend produziert. Während dieser Phase werden Daten in der Größenordnung von Terabytes (TB) pro Simulationslauf oder Petabytes (PB) pro Rechenprojekt generiert. Nach Abschluss dieser Produktionsphase schließt sich die Nachbearbeitungsphase („Postprocessing“) an, in der neue Daten anfallen, deren Größe sich noch in der Ordnung von Gigabytes befindet. Auf diese Phase folgt die Analyse der Daten, deren Ergebnisdaten sich in der Größenordnung von Megabytes befinden. Die Größe der daraus resultierenden Publikationen ist im Bereich von Kilobytes anzusiedeln. Die Publikation mit Plots, Grafiken usw. gilt selbst auch als Forschungsdatum[3]. Die hier genannten Größenordnungen sind typisch für den HPC-Arbeitsprozess, allerdings können in der Phase der Auswertung auch mehr Daten als in der Phase der Datenproduktion erzeugt werden.

1.2.3 Definition und Charakteristika von Forschungsdaten im HPC

Forschungsdaten sind gemäß der Begriffsdefinition des Rats für Informationsinfrastrukturen „Daten, die im Zuge wissenschaftlicher Vorhaben entstehen“ (Rat für Informationsinfrastrukturen, 2016, S. 11) und umfassen somit alle Daten, die für

[3] Im Bereich der Publikationen ist das Management über die Bibliotheken, Verlage, Suchmaschinen, usw. etabliert. DOIs identifizieren Publikationen eindeutig, mehrere Suchoberflächen garantieren die Findbarkeit. Anreiz zur Veröffentlichung von Ergebnissen ist die Erhöhung der eigenen wissenschaftlichen Reputation.

den wissenschaftlichen Erkenntnisprozess von Belang sind. Im HPC umfassen Forschungsdaten demnach nicht nur die Ausgabedateien der Simulationsläufe (bei GROMACS z.B. die *.trr*-Dateien), sondern auch – sofern relevant – die Zwischenergebnisse (sog. Restart-Dateien), den Simulationscode, der auf dem HPC-System ausgeführt wurde, die Job-Dateien, die den Rechenjob definieren, die Logdateien der Simulationscodes sowie die Auswertungsergebnisse und die dafür genutzten Programme. In der vorliegenden Dissertation wird der Begriff Forschungsdaten verwendet, um alle, einzelne Teile oder Kombinationen zu bezeichnen, wobei je nach Kontext erkennbar ist, wie der Begriff verwendet wird – andernfalls wird eine Erläuterung gegeben. Gemeinsame Charakteristika von Forschungsdaten im HPC lassen sich nur stichpunktartig herausarbeiten. Folgende Merkmale sind typisch:

- Die Größe von Forschungsdaten liegt im Bereich von Tera- bis Petabytes pro Rechenprojekt. Innerhalb eines Rechenprojekts sind die einzelnen Ausgabedateien der Simulationsläufe mit mehreren Giga- oder Terabytes die größten. Dies gilt es bei allen weiteren Überlegungen zu berücksichtigen. Die Größe der anderen Dateien, wie Log- und Job-Dateien sowie des Codes, sind im Bereich von Kilo- bis Megabyte anzusiedeln. Beispielsweise können Logdateien von *GROMACS* mehrere Megabyte groß werden, da je nach Einstellung sehr viele Informationen auch in die Logdateien geschrieben werden.

- Grundsätzlich sind Forschungsdaten im HPC divers. Sie stammen aus verschiedenen Quellen (Simulationscodes) und nutzen verschiedene Formate. Innerhalb eines Rechenprojekts wird zumeist ein einheitlicher Simulationscode verwendet – wobei auch hier andere Versionen oder andere Codes genutzt werden können.

- Die Relevanz der einzelnen Teile ist unterschiedlich. So wird dem Code eine hohe Wichtigkeit zugeschrieben, da hier definiert ist, wie die Daten erzeugt wurden. Ebenso sind die ausgewerteten Daten von hoher Priorität. Die Restart-Dateien sind nur innerhalb von Simulationsläufen von Bedeutung. Sie werden benötigt, um nach gewollten oder ungewollten Unterbrechungen die Simulation ab einem gewissen Zeitpunkt wieder aufnehmen zu können. Job- und Logdateien sind ebenfalls von hoher Priorität, da sie potentiell Aufschluss über die Entstehungsgeschichte der Daten geben können, also wie und mit welchen Mitteln sie erzeugt wurden.

- Zwar kommen nach jedem Simulationslauf Daten hinzu, jedoch sind die Daten, sind sie einmal erzeugt, fix und ändern sich im Gegensatz zur Code-Entwicklung nicht.

- Die Speicherung der Simulationsergebnisse erfolgt zentral auf den Speichersystemen der jeweiligen HPC-Zentren, wohingegen die Auswertungsergebnisse oft dezentral auf eigenen Speicher- und Rechnersystemen der Institute ausgeführt und gespeichert werden.

- Einmal hergestellt, sind Simulationsdaten nicht generell oder nicht ohne massiven Aufwand erneut exakt zu generieren. Die Hardware müsste konserviert und die zeitintensiven Simulationsläufe darauf wiederholt werden. Dies ist allerdings nur eine theoretische Möglichkeit, da die allgemeine Reproduzierbarkeit von Simulationsdaten durch konzeptuelle Probleme nicht gegeben ist, wie in Unterkapitel 1.3.1 ausgeführt wird.

- Forschungsdaten können je nach Phase des Forschungsprozesses, wie er in Abbildung 1.3 dargestellt ist, als *heiße*, *warme* oder *kalte* Daten kategorisiert werden. Dabei steht die Metaphorik für die Zugriffshäufigkeit auf die jeweiligen Daten und damit indirekt für die Art der Speicherung. Wird auf die Daten während der ersten und zweiten Phase noch häufig zugegriffen, so sind diese auf Festplattenspeichern oder SSD gespeichert und warm bzw. heiß. Beim Schreiben des Papers kühlen die Daten zusehends ab, um schließlich nach Beendigung des Publikationsprozesses kalt und potentiell auf Bandspeichermedien ausgelagert oder sogar gelöscht zu werden.

1.3 Herausforderungen von Forschungsdatenmanagement im HPC

Im vorliegenden Unterkapitel findet eine Beschreibung der Herausforderungen für Forschungsdatenmanagement in den Simulationswissenschaften statt, welche ich in technische, organisatorische und rechtliche Herausforderungen gliedere. Technische Herausforderungen stellen Probleme dar, die technischer Natur sind, wie z.B. die großen Datenmengen im HPC. Organisatorische Herausforderungen rühren von fehlenden Prozessen oder deren mangelhafter Organisation her, wie z.B. die Frage nach der Datenverantwortung. Rechtliche Herausforderungen betreffen alle rechtlichen Fragen, wie z.B. die des Datenschutzes. Da sich einzelne Herausforderungen auch in mehrere der drei Kategorien einordnen lassen, findet sich am Ende des Unterkapitels eine Zusammenfassung, in der noch einmal alle Herausforderungen diskutiert werden. Die in diesem Unterkapitel dargestellten Herausforderungen führen – direkt oder indirekt – zur Entstehung dunkler Daten, die in Kapitel 3 diskutiert wird.

1.3.1 Technische Herausforderungen

Die technischen Herausforderungen für Forschungsdatenmanagement im HPC sind auch unter *Volume* und *Variety* im Zusammenhang mit dem Begriff Big Data bekannt (Schembera und Bönisch, 2017) und werden nun näher analysiert.

Das Volumen von Forschungsdaten liegt pro HPC-Zentrum derzeit bei mehreren Petabytes, da pro Rechenprojekt zumeist mehrere Terabytes an Daten anfallen. Eine Studie (Hick, 2010) des Energieministeriums der Vereinigten Staaten, dem größten Betreiber von HPC-Systemen in den USA, kommt durch empirische Analyse zum

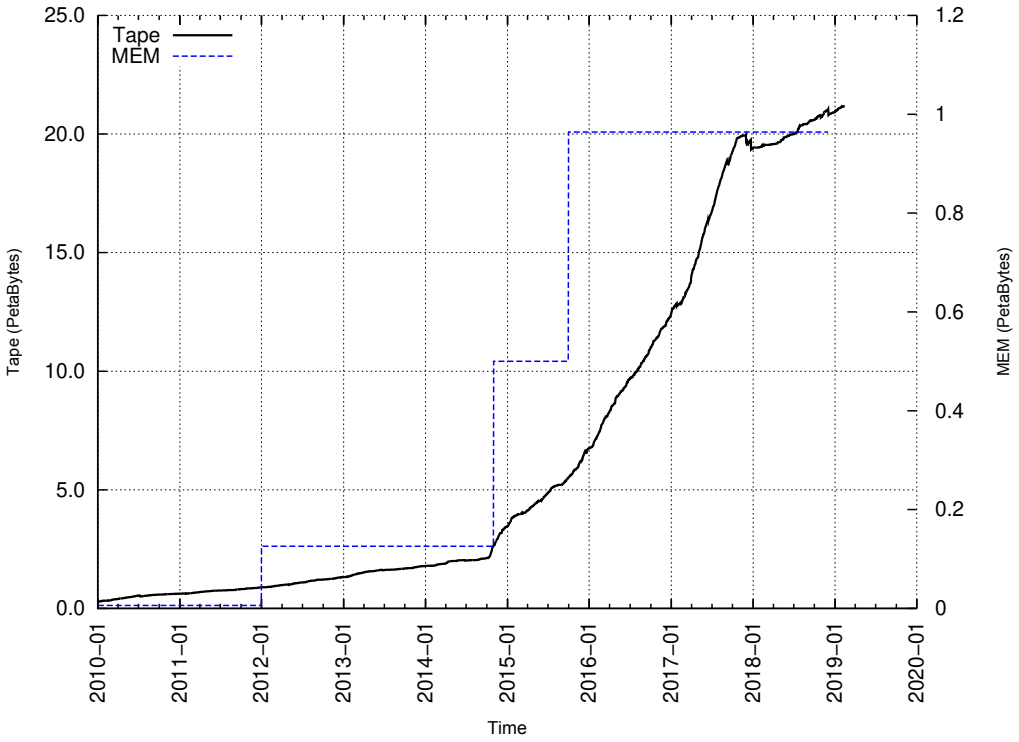

Abbildung 1.4: Verhältnis des Hauptspeichers (blaue, gestrichelte Linie) gegenüber der Belegung des Bandspeichersystems (schwarze, durchgehende Linie) am HLRS von 2010-2019. Im Februar 2019 wurden ca. 21 PB an Daten auf zwei Kopien im Bandspeichersystem gehalten. Die zu diesem Zeitpunkt aktuelle Cray XC 40 „Hazel Hen“ verfügt über einen Hauptspeicher von 984 TB.

Ergebnis, dass 1 Byte Hauptspeicher ca. 35 Bytes Archivdaten pro Jahr produzieren wird, wobei 20-50 % wieder gelöscht werden. Das Volumen der produzierten Daten ist also allein von der Größe des Hauptspeichers abhängig, nämlich im bestimmten Verhältnis von 35:1 (Daten:Speicher). Grafik 1.4 zeigt exemplarisch die Speicherbelegung des Bandarchivsystems gegenüber dem Hauptspeicher des zum jeweiligen Zeitpunkt aktuellen Großrechners am HLRS von 2010-2019. Im Februar 2019 wurden am HLRS ca. 21 PB an Daten auf zwei Kopien im Bandarchivsystem gehalten bei einem Hauptspeicher von 984 TB des aktuellen Großrechnersystems Cray XC40. Es zeigt sich, dass das jährliche Verhältnis weit unter dem der Studie liegt, jedoch tendenziell der Archivspeicher immer ein Vielfaches des Arbeitsspeichers des jeweiligen Hauptsystems ausmacht. Bezüglich der Zahl der Dateien konnten in der Studie keine Schätzwerte abgegeben werden, da diese durch Dateiaggregation verfälscht ist. Jedoch kann davon ausgegangen werden, dass die Zahl der Dateien von der Menge der Prozessoren und Cores in einem HPC-System abhängt.

Die Probleme, die mit dem Volumen der Daten einhergehen, sind deren langfristige Speicherung auf sicheren Trägern sowie deren Verschiebbarkeit zwischen den Standorten, auch über Institutsgrenzen hinweg. Mehrere Petabytes auf zweien oder mehreren Kopien über Jahre zu halten ist ein teures Unterfangen. Mehrere Petabytes

```
[st_ac108829@uc1n996 run]$ pwd
/home/st/st_st/st_ac108829/simulations/binary/educt_hexane/300_020_080/run
[st_ac108829@uc1n996 run]$ ls -alrth
insgesamt 3,3G
-rw-r--r--  1 st_ac108829 st_st  423  4. Jan 11:15 run.job
-rw-r--r--  1 st_ac108829 st_st  12K  4. Jan 11:16 run.mdp
-rw-r--r--  1 st_ac108829 st_st  31M  4. Jan 11:16 run.log
-rw-r--r--  1 st_ac108829 st_st  358  4. Jan 11:16 run0.job
-rw-r--r--  1 st_ac108829 st_st 2,8K  4. Jan 11:16 box.gro
drwxr-xr-x 10 st_ac108829 st_st 4,0K  4. Jan 11:16 ..
-rw-r--r--  1 st_ac108829 st_st 3,3G  4. Jan 11:18 run.trr
drwxr-xr-x  2 st_ac108829 st_st 4,0K  4. Jan 11:19 .
```

Abbildung 1.5: Screenshot einer Dateiorganisation von GROMACS-Simulationsdaten auf dem bwUniCluster. Dabei gibt das Verzeichnis wichtige Informationen an, nämlich, dass es sich um eine binäre Mischung mit dem Ausgangsstoff Hexan in einem Mischverhältnis 20 % zu 80 % handelt und die Temperatur 300K beträgt. Die Dateien im Verzeichnis sind die Job- (.job), Log-(.log oder .mdp), Topologie- (.gro) und Ergebnis- bzw. Trajektoriedateien (.trr).

im Rahmen eines Technologiewechsels auf neue Bandspeichertechnologie zu migrieren ist zeitintensiv. Es ist ebenso nicht mehr möglich, diese in angemessener Zeit über Rechnernetze an einen anderen Standort zu bringen. Eine Folgestudie des gleichen Autors (Hick, 2013) kommt zu dem Schluss, dass sich die Lage im Exascale-Zeitalter verschärfen wird, insbesondere weil eine Lücke bzgl. verfügbarer Speichertechnologie besteht. Darüber hinaus ist es ein grundsätzliches Problem, bei Tausenden Dateien, die in einer Verzeichnisstruktur organisiert sind, den Überblick zu behalten.

Die Vielfalt (variety) der Daten verschärft das Problem, denn die Daten kommen aus verschiedenen Quellen, bestehen aus diversen Dateien und unterliegen unterschiedlichen Namenskonventionen (Arora, 2015; Jones u. a., 2011; Parker-Wood u. a., 2013). Selbst wenn innerhalb einer Forschungsgruppe mit einem Simulationscode gearbeitet wird, ist die Namensgebung für Dateien und Verzeichnisse oft unterschiedlich und hängt stark von der Institutskultur ab. Metadaten werden, wenn überhaupt, nur implizit gespeichert, nämlich über die Verzeichnisstrukturen und Dateinamen, anhand derer sich Informationen über die Daten ableiten lassen. Eine typische Organisationstruktur für Simulationen ist hier dargestellt:

```
/group/project/user/simulation/run/description.format
```

Sie zeigt die hierarchische Gliederung der Verzeichnisse und den Dateinamen sowie die implizite Natur der Beschreibung auf dieser Ebene. Ein Beispiel einer Verzeichnisorganisation von Forschungsdaten aus der Thermodynamik, die mit GROMACS erzeugt wurden, sowie der Verzeichnisinhalt mit den Job-, Log-, Topologie- und Ergebnisdateien findet sich in Abbildung 1.5. Da auch in den meisten Dateien keine weiteren expliziten – und falls doch, hauptsächlich technische – Metadaten enthalten sind, ist die Organisation der Daten maßgeblich erschwert. Sind keine expliziten, allgemeingültigen, allgemein verständlichen und standardisierten Beschreibungen von Daten technischer, fachspezifischer, allgemeiner oder prozessspezifischer Art vorhanden, lassen sich die Daten nur verstehen, wenn implizites Wissen vorliegt.

Dies besitzen meist nur die Erzeugenden oder die direkt am Entstehungsprozess Beteiligten. (Hick, 2013) zufolge bedarf es revolutionärer Ansätze, um diese Lücke zu schließen. Eine Analyse (Iglezakis und Schembera, 2018) der Ergebnisse einer Umfrage an allen baden-württembergischen Universitäten (Tristam und Streit, 2015)[4], die im Rahmen des Projekts bwFDM-Communities (Pappenberger, 2016) durchgeführt wurde, zeigt, dass in den Ingenieurwissenschaften insbesondere Unterstützungsbedarf bei der Beschreibung von Daten, also der Metadatenannotation besteht.

Neben allen technischen Herausforderungen ist die aufwendige Reproduzierbarkeit von Simulationsdaten auf der Bit-Ebene eine zusätzliche Motivation für geregeltes Forschungsdatenmanagement. Reproduzierbarkeit auf Bit-Ebene kann nur durch massive Performance-Verluste erkauft werden, so dass im Normalfall also von Nicht-Reproduzierbarkeit ausgegangen werden sollte. Diese Nicht-Reproduzierbarkeit hat ihren Ursprung einerseits in den der Simulationsmethode inhärenten statistischen und systematischen Fehlern. Ein Beispiel hierfür ist der Nicht-Determinismus durch dynamischen Lastausgleich oder durch Compileroptimierungen. Andererseits können die Gründe auf algorithmische Fehler, Softwarefehler und Implementierungsprobleme, aber auch auf Hardwarefehler und -änderungen zurückgeführt werden. Da in heutigen Höchstleistungsrechnern Tausende Prozessoren und Speichermodule verbaut sind, steigt auch die Wahrscheinlichkeit für Hardwaredefekte (Ludwig und Geyer, 2019). Die Nicht-Reproduzierbarkeit beschränkt sich nicht nur auf die Bit-Ebene. So kommt eine Studie (Schappals u. a., 2017) zu dem Ergebnis, dass selbst ein und derselbe thermodynamische Simulationscode von verschiedenen Arbeitsgruppen auf unterschiedlichen Rechnern mit gleichen Startkonfigurationen und Parametern ausgeführt verschiedene Ergebnisse liefert, die nicht auf statistisch normale Abweichungen zurückzuführen sind, sondern systemische Probleme enthalten. Damit sei der Begriff der Reproduzierbarkeit, bezogen auf Simulationen, wesentlich problematischer als bezogen auf Experimente (Resch und Kaminski, 2019). Eine ebenso große Varianz der Ergebnisse wurde in einer Studie für die Astrophysik ermittelt. Dabei wurde eine Simulation zur Ermittlung von Masse, Dichte und Position von Himmelsobjekten 200 Mal mit identischen Startbedingungen und -eigenschaften durchgeführt, was zu 18 verschiedenen Ergebnissen über die Masse und 16 verschiedenen Ergebnissen bzgl. der X-Koordinate führte (Stodden, 2018). Diese Studie wurde im Rahmen der *Reproducibility Challenge*[5] durchgeführt, die im Rahmen der größten Konferenz im Bereich Höchstleistungsrechnen, der *Supercomputing Conference*, im Jahr 2018 stattfand[6]. In (Fehr u. a., 2016) und (Ludwig und Geyer, 2019) wird auch von einer *Reproduzierbarkeitskrise* gesprochen: Zwar sollte an Wissenschaft der Anspruch

[4] Die Ergebnisse der Studie sind auch online zugreifbar: `http://bwfdm.scc.kit.edu/cgi-bin/daten/`, Zugriff 4.6.2019.

[5] `http://rescue-hpc.org/program2018.html`, Zugriff 22.2.2019.

[6] Im Rahmen dieser *Reproducibility Challenge* wird Studierenden für die Konferenz 2019 die Aufgabe gegeben, die Ergebnisse eines für die Konferenz 2018 akzeptierten Papers zu reproduzieren (`https://sc19.supercomputing.org/submit/reproducibility-initiative/`, Zugriff 22.2.2019).

gestellt werden, dass sich Ergebnisse bei gleicher Startkonfiguration reproduzieren lassen, allerdings ist dieser Anspruch in der Praxis, speziell bei der Simulation mittels paralleler Methoden, nicht im eigentlichen Sinne einzulösen.

1.3.2 Organisatorische Herausforderungen

Forschungsdatenmanagement im HPC steht nicht nur vor technischen, sondern auch vor organisatorischen Herausforderungen. Damit sind alle Probleme gemeint, die aus der Organisation der Arbeitsprozesse bezüglich des Managements von Daten herrühren. In der Studie *Do researchers dream of research data management?* (Wilms u. a., 2018) wird festgestellt, dass nicht-technische Anforderungen an FDM den Hauptgrund für Hürden darstellen, weswegen Wissenschaftlerinnen und Wissenschaftler FDM nicht richtig betreiben können, also definitiv nicht von Forschungsdatenmanagement (oder von elektrischen Schafen) träumen.

Im HPC sind die größten Probleme die fehlenden Anreize zum zielgerichteten, auf Nachnutzbarkeit der Daten ausgerichtetes Datenmanagement (Iglezakis und Schembera, 2018). Im Mittelpunkt der wissenschaftlichen Arbeit steht die Veröffentlichung einer wissenschaftlichen Publikation, die als Maß für die Güte von Wissenschaftlerinnen und Wissenschaftlern gilt (Petersen u. a., 2014). Dies schafft Anreize, die Veröffentlichung selbst entsprechend zu dokumentieren und mit Metadaten, wie Informationen über die Autorinnen und Autoren, zu versehen. Die Dokumentation, Ablage und evtl. Veröffentlichung von Daten, die zu dieser Publikation geführt haben, bedeutet jedoch nur einen Zusatzaufwand, der nicht in die wissenschaftliche Reputation eingeht, sondern höchstens zum ungeliebten Handwerk gehört (Bruce und Hillmann, 2004). Oft ist der Nutzen nicht klar, Daten überhaupt zu speichern. Dies ist speziell im HPC ein umstrittenes Thema, bei dem sich Möglichkeiten zur Reproduzierbarkeit, Kosten/Nutzen bei Speicherung von sehr großen Datenmengen und deren Wichtigkeit schneiden.

Darüber hinaus resultieren die Herausforderungen auch aus fehlenden Ausbildungsmaßnahmen zum Thema Datenmanagement. Einer Studie zufolge (Cox und Pinfield, 2014) sind fehlende Fähigkeiten und Fertigkeiten im Bereich Datenmanagement eine der größten Herausforderungen, die gelöst werden müssen. Ebenso spricht Chris Mattmann in einem Artikel in *Nature* (Mattmann, 2013) davon, dass eine neue Generation Wissenschaftlerinnen und Wissenschaftler antreten müsse, die sich auch als DatenwissenschaftlerInnen verstehe und Fähigkeiten in diesem Bereich hätte, d.h. auch die Datenverarbeitung und die langfristige Datenpflege beherrsche.

Im gleichen Artikel (Mattmann, 2013) wird das Problem der Datenverantwortung thematisiert, das auch zu den organisatorischen Herausforderungen gehört. Die Verantwortung über einmal produzierte Forschungsdaten ist oft unklar, insbesondere in der dynamischen akademischen Welt. Im besten Fall wird eine Person an einem Institut in die Rolle des- oder derjenigen gedrängt, der/die sich um das Datenhandling kümmert. Dies ist aber keine definierte Rolle wie die eines Datenschutzbeauftragten,

sondern wird eher nebenher erledigt. Scheiden nun Mitarbeiterinnen oder Mitarbeiter aus, oder Doktorandinnen oder Doktoranden verlassen das Institut, ist oft nicht klar, wie mit den Daten weiter verfahren werden soll – die Datenverantwortlichkeit ist unklar. Daten können dadurch nicht mehr zugreifbar sein, worauf in Kapitel 3 gesondert eingegangen wird.

Das Problem der Datenverantwortung verschärft sich noch durch nachlässige Planung und fehlende Datenmanagementpläne. Datenmanagementpläne definieren Verantwortlichkeiten, Zeiträume der Speicherung und das Dokumentationsmodell. Obwohl diese zunehmend von den Geldgebern gefordert werden (DFG, 2018; EU, 2016b; NSF, 2014), gibt es wenig Engagement in diesem Bereich (Schembera und Bönisch, 2017).

1.3.3 Rechtliche Herausforderungen

Auch Forschungsdaten können im Rahmen des Urheberrechtsgesetzes (UrhG) relevant sein, allerdings liegen die Probleme eher im datenschutzrechtlichen Bereich (Spindler und Hillegeist, 2011). Da im HPC mit erzeugten, aber weniger mit personenbezogenen Daten umgegangen wird als z.B. in der Medizininformatik, ist dieser Themenkomplex von geringer Relevanz für das Dissertationsvorhaben und wird nicht weiter ausgeführt. Die Leserinnen und Leser seien auf den Artikel „Rechtliche Probleme der elektronischen Langzeitarchivierung von Forschungsdaten“ (Spindler und Hillegeist, 2011) im *Handbuch Forschungsdatenmanagement* (Büttner, Hobohm und Müller, 2011) verwiesen, ebenso auf das *Gutachten zu den rechtlichen Rahmenbedingungen des Forschungsdatenmanagement* (Lauber-Rönsberg, Krahn und Baumann, 2018), das im Rahmen des Projekts DataJUS[7] erstellt wurde.

1.3.4 Diskussion der Herausforderungen

In den vorhergehenden Unterkapiteln wurde eine Kategorisierung in technische, organisatorische und rechtliche Herausforderungen unternommen. Diese Einteilung zeigte sich ex post als passend, weil im Rahmen der Dissertation einerseits technische und andererseits organisatorische Lösungsvorschläge entwickelt wurden. So wird diese Trennung in der weiteren Arbeit beibehalten.

Gewisse Themen haben allerdings mehrere Dimensionen. So lassen sich die Herausforderungen nach Metadatenannotation nicht allein technisch einordnen. Neben der Nichtverfügbarkeit eines Metadatenmodells fehlen – wie bereits ausgeführt wurde – auch Anreize, Metadatenannotation durchzuführen. Ebenso hat die vornehmlich organisatorische Herausforderung der Datenverantwortung auch eine technische Dimension: Wie werden nicht mehr zugeordnete Daten technisch ermittelt, oder wie werden sie einer dritten, zuständigen Person überantwortet?

[7] `https://tu-dresden.de/gsw/jura/igetem/jfbimd13/forschung/forschungsprojekt-datajus`, Zugriff 1.4.2019.

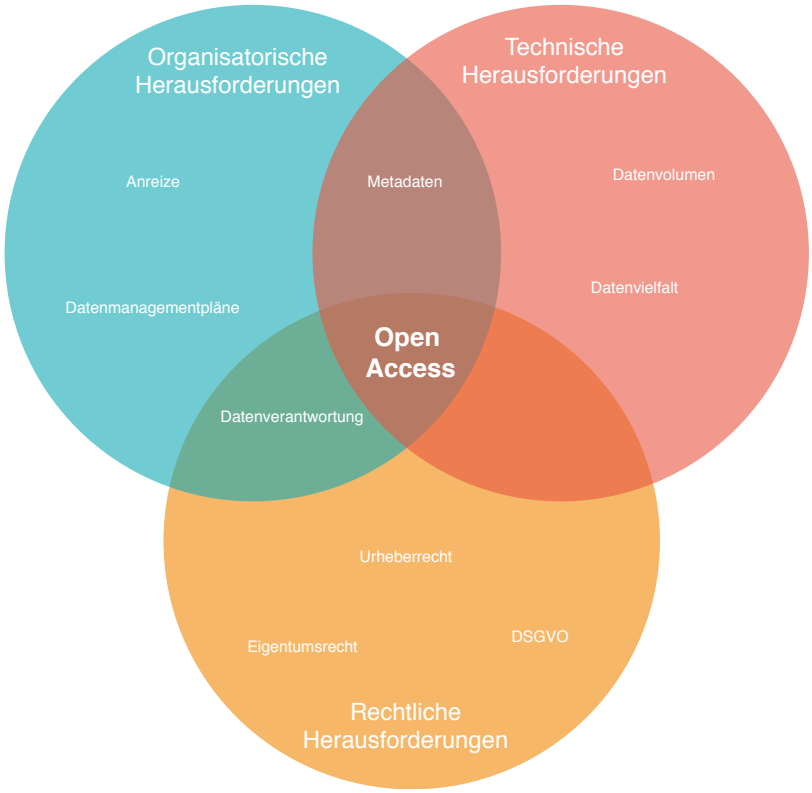

Abbildung 1.6: Darstellung der Herausforderungen von Forschungsdatenmanagement im HPC als Venn-Diagramm. Dabei vereinen sich rechtliche, organisatorische und technische Herausforderungen im Open Access Paradigma.

Ebenso gibt es eine Herausforderung, die genau an der Schnittfläche von technischem, organisatorischem und rechtlichem liegt: Open Access. Open Access gilt als das Paradigma, Textpublikationen, aber auch öffentlich finanzierte Forschungsergebnisse, zu denen insbesondere auch Daten gezählt werden, öffentlich, bestenfalls in zentralen Repositorien, frei zugänglich zur Verfügung zu stellen. Dieses Paradigma wird (Cox und Pinfield, 2014) zufolge in Zukunft zur obersten Priorität werden. Die EU fordert die Bereitstellung von Forschungsdaten von zukünftigen Projekten als Open Access bereits ein (EU, 2016a). Die DFG sieht Open Access als Vision für zukünftiges Datenmanagement (DFG, 2013). Open Access soll redundante Arbeit verhindern, dadurch Ressourcen sparen, ebenso Überprüfbarkeit und die gute wissenschaftliche Praxis ermöglichen sowie die Verschiebbarkeit von Daten garantieren. Während Open Access bei Textpublikationen große Fortschritte macht[8], ist es bezogen auf Daten weit von Akzeptanz entfernt. In einer Studie wurde gezeigt, dass trotz vorhandener Plattformen und Infrastrukturen nur 10 % der angeschriebenen Autorinnen und Autoren von Veröffentlichungen auf PLOS[9] auf die Nachfrage nach den Forschungsdaten überhaupt antworteten (Wilms u. a., 2018). Es fehlen Anreize

[8] Siehe bspw. das DEAL-Projekt: `https://www.projekt-deal.de/`, Zugriff 1.4.2019.
[9] `https://www.plos.org/`, Zugriff 23.4.2019.

für Wissenschaftlerinnen und Wissenschaftler, Daten aufzubereiten und bereitzustellen, da die Reputation hierfür gering ist. Darüber hinaus reicht Open Access in die technischen Herausforderungen hinein: Transfertechnologien zur Verschiebung sehr großer Datenmengen sind nicht verfügbar, ebenso keine Metadatenmodelle im Bereich HPC. Rechtliche Fragen wie die Klärung des Urheberrechts bleiben weiterhin offen.

Die hier diskutierten Verknüpfungen zwischen den technischen, organisatorischen und rechtlichen Herausforderungen sind in Abbildung 1.6 in einem Venn-Diagramm dargestellt. Sie vereinen sich im Open Access Paradigma, wonach die öffentliche Bereitstellung von Forschungsdaten im Rahmen von HPC und Simulationen eine besondere Herausforderung darstellt, da hier einige Charakteristika, wie z.B. die sehr großen Datenmengen, besonders brisant werden.

1.4 Problemstellung

Wie gezeigt wurde, steht Forschungsdatenmanagement im HPC vor Herausforderungen, die Daten zur Folge haben, die nicht FAIR sind.

Demgegenüber steht die Vorstellung vom *Zweck* des Forschungsdatenmanagements, die Daten FAIR, d.h. findbar, zugreifbar, interoperabel und nachnutzbar zu halten. Dabei müssen die Daten über den ganzen Lebenszyklus, auch über das Projektende hinaus, zugänglich gehalten werden. Dafür muss Forschungsdatenmanagement insbesondere technische und organisatorische *Mittel* zur Verfügung stellen. Auf der technischen Seite müssen die, in Kapitel 1.3.1 diskutierten, Herausforderungen der fehlenden Metadaten durch mangelnde Beschreibungsmöglichkeiten und die kaum vorhandene Prozessintegration gelöst werden. Auf der Seite organisatorischer Herausforderungen, die in Kapitel 1.3.2 besprochen wurden, muss die Prozessintegration nicht-technisch unterstützt werden und dunkle Daten müssen verhindert werden. Dunkle Daten widersprechen genau der Definition von FAIRen Daten, da dunkle Daten weder findbar oder zugreifbar, noch interoperabel oder nachnutzbar sind. Dies wird in Kapitel 3 dargestellt.

1.5 Forschungsbeitrag

Im Rahmen der Dissertation wurden zunächst die Herausforderungen im Bereich des HPC expliziert, womit eine Kategorisierung in technische und organisatorische einherging. Sich aus diesen Herausforderungen ergebend, wurde der Begriff der dunklen Daten konkret auf HPC bezogen – mit dem Ergebnis, dass sich auch in diesem Feld dunkle Daten bilden können. Dafür wurde im Rahmen der Dissertation eine auf HPC und seine Speicherhierarchie maßgeschneiderte Definition für dunkle Daten entwickelt. Die Existenz dunkler Daten und deren zeitlicher Verlauf wird durch Statistiken belegt, die über das Bandspeichersystem des HLRS erstellt wurden. Ebenso wurde speziell auf die Implikationen dunkler Daten im Kontext

des Höchstleistungsrechnens eingegangen und gezeigt, warum dunkle Daten dem gegenwärtigen Datenlebenszyklus inhärent sein müssen.

Auf den Herausforderungen aufbauend, werden Anforderungen formuliert, die zu Konzepten für Forschungsdatenmanagement im Kontext dunkler Daten in den Simulationswissenschaften führen. Dies bedeutet, dass Lösungen erarbeitet werden, um dunkle Daten im Höchstleistungsrechnen entweder zu vermeiden oder zu verringern. Dies ist insbesondere ein Metadatenmodell für die Thermo- und Aerodynamik, zwei Hauptdisziplinen der Ingenieurwissenschaften, die enorm von HPC profitieren. Dieses Metadatenmodell *EngMeta* enthält domänen- und fachspezifische Merkmale und ist die Basis für ein automatisiertes Metadatenannotationskonzept, das zur automatisierten Metadatenerfassung entwickelt und implementiert wurde. Diese Entwicklungen wurden in ein konzeptuelles, dreischichtiges Systemmodell integriert. Schließlich wurden noch Entscheidungskriterien entwickelt, die die Wichtigkeit der Daten gegenüber deren Speicherungbedarf abwägt. Darüber hinaus wurde die Rolle des Scientific Data Officer, einem spezifischem Datenkurator für HPC ausformuliert, der – ähnlich einem Datenschutzbeauftragten für rechtliche Fragen – für alle technischen und organisatorischen Belange des Datenmanagements zuständig ist.

Einige Konzepte dieser Dissertation wurden im Rahmen von Projekten oder in sonstigen Kollaborationen erarbeitet und in Publikationen vorab veröffentlicht, was hier und in den jeweiligen Kapiteln gekennzeichnet und aufgeschlüsselt ist.

Dem Projekt DIPL-ING (Datenmanagement in Infrastrukturen, Prozessen und Lebenszyklen in den INGenieurwissenschaften)[10] entspringt die Anforderungsanalyse für die Ingenieurwissenschaften, die in (Iglezakis und Schembera, 2018) veröffentlicht wurde. Dabei wurden vom Autor insbesondere die HPC-spezifischen Anforderungen eingebracht, die hier in Kapitel 4 in stark überarbeiteter Form eingehen. Ebenso innerhalb DIPL-ING wurde das Metadatenmodell *EngMeta* zusammen mit Dorothea Iglezakis (UB) und Gernot Bauer (ITT) basierend auf den Anforderungen, die von den ingenieurwissenschaftlichen Instituten qualitativ erhoben wurden, entwickelt. Dieses wurde im Tagungsband der *International Conference on Metadata and Semantics Research* (MTSR)-Konferenz 2018[11] (Schembera und Iglezakis, 2019) publiziert. Dabei brachte Gernot Bauer die fachwissenschaftliche Perspektive und Dorothea Iglezakis die Perspektive zu Metadatenstandards und -vokabularen ein. Der Autor der Dissertation brachte insbesondere die HPC- und Informatik-Perspektive in die Entwicklung des Metadatenmodells ein. Die Evaluation des Metadatenmodells sowie der Erfassung und Arbeitsprozessintegration wurde allein vom Autor im Rahmen der Dissertation durchgeführt und findet sich in Kapitel 6.2. Die Metadatenerfassung wurde ausschließlich vom Autor selbst vorgeschlagen, entworfen und implementiert. Lediglich die Anforderungen an die Applikation wurden innerhalb des Projektes mit den Fachwissenschaftlerinnen und -wissenschaftlern ermittelt und dann vom

[10] `http://www.ub.uni-stuttgart.de/forschen-publizieren/forschungsdatenmanagement/projekte/dipl_ing/index.html`, Zugriff 4.7.2018.

[11] `http://www.mtsr-conf.org/2018/home`, Zugriff 10.5.2019.

Autor umgesetzt. Teile der Dissertation aus Kapitel 5.2 und Kapitel 5.3 wurden für die Abschlusspublikation des Projekts DIPL-ING verwendet, welche für den Tagungsband der eScience-Tage 2019 in Heidelberg[12] vorgesehen ist und sich momentan in Begutachtung befindet. Da die Metadatenerfassung zukünftig auch vom Sonderforschungsbereich (SFB) 1313[13] der Universität Stuttgart genutzt werden soll, wurde die Anpassung auf den spezifischen Anwendungsfall in einer allgemeinen Publikation zur Planung des Datenmanagements in diesem SFB in der Fachzeitschrift *Bausteine FDM*[14] eingereicht. Ebenso sind Inhalte der Dissertation aus den Kapiteln 5.2, 5.3 und 6.2 zur Publikation einer erweiterten Fassung des Konferenzbeitrags (Schembera und Iglezakis, 2019) im *International Journal of Metadata, Semantics and Ontologies*[15] vorgesehen, zu der die beiden Autoren vom Programmkommittee der MTSR-Konferenz eingeladen wurden.

Die Arbeit am Begriff der dunklen Daten und dem Scientific Data Officer stammt aus einer gemeinsamen Vorarbeit mit Juan Duràn, die im Paper (Schembera und Durán, 2019) veröffentlicht wurde. Dabei erarbeitete Juan Duràn insbesondere die ethischen Implikationen von dunklen Daten sowie die ethischen Verantwortlichkeiten des Scientific Data Officers und trug zur technikphilosophischen Ausgestaltung der Begrifflichkeiten und der Rolle bei. Auf dieser Vorarbeit aufbauend wurden vom Autor für die Dissertation erweiterte Statistiken über dunkle Daten im zeitlichen Verlauf am HLRS erstellt. Außerdem wurden die dunklen Daten durch fehlende Metadaten stärker thematisiert, da sie den eigentlichen Hauptgrund für dunkle Daten darstellen, wie sich während der Analyse zeigte. Darüber hinaus wurde im Rahmen der Dissertation vom Autor eine enge Definition für dunkle Daten im HPC vorgelegt sowie stärker auf die Implikationen von dunklen Daten eingegangen. Ebenso wurde in der Dissertation hergeleitet, warum dunkle Daten nicht FAIR sind und dem heutigen Datenlebenszyklus inhärent sein müssen. Bezogen auf den Scientific Data Officer finden sich die Teile genau im entsprechenden Kapitel 5.4 aufgeschlüsselt.

1.6 Methodologie

Die Dissertation setzt in ihrer Methodologie auf die Prinzipien des Software Engineering (Ludewig und Lichter, 2013), die sich nach den Phasen *Anforderungsanalyse und Spezifikation*, *Architekturentwurf und Konzept*, *Ausprogrammierung* und *Validierung* richtet. Zwei Anwendungsfälle dienen der Problemexplikation und sind die Ausgangspunkte für die weitere Arbeit, indem sie die funktionalen Anforderungen an Forschungsdatenmanagement aus der Außensicht heraus vorgeben. Die beiden Anwendungsfälle sind die computergestützte Modellierung in *Thermodynamik* und *Aerodynamik*. Die Anwendungsfälle wurden ausgewählt, weil sie repräsentativ für computergestützte Ingenieurwissenschaften sind, die mit Hilfe von HPC-Infrastruktur

[12] `https://e-science-tage.de/de/startseite`, Zugriff 10.5.2019.
[13] `https://www.sfb1313.uni-stuttgart.de/de`, Zugriff 10.5.2019.
[14] `https://bausteine-fdm.de/`, Zugriff 20.5.2019.
[15] `https://www.inderscience.com/jhome.php?jcode=ijmso`, Zugriff 10.5.2019.

wissenschaftliche Erkenntnis produzieren. Beide nutzen dabei den in Kapitel 1.2.2 dargestellten Arbeitsprozess und produzieren dabei Forschungsdaten, die im Wesentlichen den Charakteristika aus Kapitel 1.2.3 entsprechen: große bis sehr große Daten im Bereich von Tera- bis Petabytes, Diversität der Datenformate, unterschiedliche Wichtigkeit der einzelnen Dateien, Speicherung der Daten auf dem parallelen Dateisystem oder in einem Bandspeichersystem ohne Repositoriumsunterstützung, kritische Reproduzierbarkeit der Daten. Die beiden Anwendungsfälle werden im Folgenden besprochen, ebenso die jeweilig genutzten Datenformate.

1.6.1 Anwendungsfall 1: Thermodynamik

Die Thermodynamik befasst sich u.a. mit der computergestützten Modellierung von Molekülen und ihren Bewegungen. Dabei werden Höchstleistungsrechner und Clustersysteme benutzt, um die Trajektorien der Moleküle zu berechnen. Die konkreten Beispieldaten entstammen dem *Institut für Technische Thermodynamik und Thermische Verfahrenstechnik*[16] (ITT) der Universität Stuttgart und wurden auf dem bwUniCluster[17] mit dem Simulationscode GROMACS erzeugt. Dabei ist es das Ziel, eine möglichst umfassende Beschreibung der Daten zu erhalten, sowie eine automatische Erfassung der Metadaten für die einzelnen Simulationsläufe zu ermöglichen. Idealerweise sollen die Daten automatisiert in ein Repositorium gespielt werden können.

Das in diesem Anwendungsfall verwendete Paket GROMACS (Groningen Machine for Chemical Simulations) ist ein Simulationscode für die Molekulardynamik[18] (Hess u. a., 2013). Der typische Arbeitsprozess bei einer Simulation mit GROMACS ist in Abbildung 1.7 dargestellt. Nachdem die Topologieinformation generiert wurde und in einer *.gro*- oder *.top*-Datei zur Verfügung steht, werden mittels des GROMACS Preprocessors *grompp* alle für den eigentlichen Simulationslauf benötigten Dateien erzeugt. Dieser wird dann mit *gmx mdrun* gestartet und erzeugt die eigentlichen Simulationsdaten in Form von Trajektorien. Die Informationen über die Trajektorien der Moleküle können dabei von GROMACS in verschiedene Ausgabedateien geschrieben werden, wobei insbesondere die binäre *.trr*-Datei interoperabel alle Informationen enthält. Im Gegensatz zum EAS3-Format findet sich in der binären Datei selbst kein lesbarer Kennsatz. Prozess- und fachspezifische Metadaten finden sich in den GROMACS-Logdateien mit der Endung *.log*, sowie in den weiteren, am Prozess beteiligten Dateien wie der Topologiedatei. Grundsätzlich sind bei GROMACS alle Metainformationen über mehrere Dateien verteilt.

[16] `https://www.itt.uni-stuttgart.de/`, Zugriff 2.3.2019.

[17] `https://www.bwhpc-c5.de/wiki/index.php/Category:BwUniCluster`, Zugriff 13.1.2019.

[18] `http://www.gromacs.org/`, Zugriff 28.2.2019

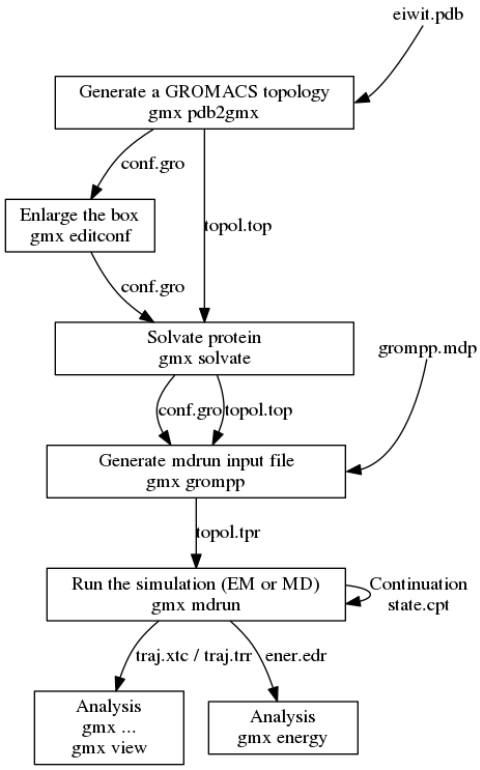

Abbildung 1.7: Typischer Arbeitsprozess bei einer Simulation mit GROMACS. Quelle: `http://manual.gromacs.org/documentation/2018.3/user-guide/flow.html`, (Zugriff 1.3.2019).

1.6.2 Anwendungsfall 2: Aerodynamik

Der zweite Anwendungsfall entstammt dem *Institut für Aerodynamik und Gasdynamik*[19] (IAG) der Universität Stuttgart und umfasst Daten aus Strömungssimulationen. Die Daten aus diesem Anwendungsfall wurden auf der Großrechneranlage des HLRS, der Cray XC40 „Hazel Hen“ mit Computational Fluid Dynamics (CFD)-Simulationscodes[20] generiert und werden im am IAG entworfenen Datenformat *eas3*[21], abgelegt. Zusätzlich werden Metainformationen zur Simulation auch in die Datei *output.log* geschrieben. Hierbei soll Forschungsdatenmanagement als Hilfsmittel dienen, Datendokumentation – auch für interne Zwecke – zu ermöglichen und Metadaten automatisch zu erfassen. Dadurch soll der Überblick über sehr viele und umfangreiche Daten mit einem großen Parameterraum behalten werden. Es soll auch möglich sein, sehr große Datenmengen automatisiert in ein Repositorium zu spielen.

EAS3 (Ein-Ausgabe-System 3) ist ein Dateiformat für die Aerodynamik, das am Institut für Aero- und Gasdynamik der Universität Stuttgart entwickelt wurde. Es stellt dabei ein standardisiertes Binärformat nach IEEE-Norm für Gleitkommazahlen

[19] `https://www.iag.uni-stuttgart.de/`, Zugriff 2.3.2019

[20] Dabei wird u.a. der Simulationscode FLOWer genutzt (`https://www.dlr.de/as/desktopdefault.aspx/tabid-395/526_read-694/`, Zugriff 8.5.2019).

[21] `https://wiki.iag.uni-stuttgart.de/eas3wiki/index.php/Main_Page/de`, Zugriff 13.1.2019.

dar, um Daten zwischen verschiedenen Rechnerarchitekturen (wie z.B. Little- oder Big-endian) austauschen zu können. Hierfür können in die CFD-Simulationscodes Bibliotheken eingebunden werden, die Daten im EAS3-Format schreiben bzw. lesen können, womit die Interoperabilität der Daten über Rechnerarchitekturen hinweg gewährleistet wird. In diesen Dateien gibt es am Dateibeginn jeweils einen Kennsatz[22], der einige Metadaten enthält, wie z.B. die Genauigkeit, die Anzahl der Zeitschritte sowie Parameter oder die Anzahl der Gitterpunkte in jede Raumrichtung. Gefolgt von diesem Kennsatz finden sich dann die eigentlichen in der Simulation erzeugten Daten. Diese Daten sind als Parameter organisiert, wobei diese Felder mit Fließkommazahlen darstellen und sich in Zeitschritte gruppieren lassen. EAS3 ist also ein Containerformat, welches die Daten mitsamt einiger Metadaten als Kennsatz in einer Datei vereint speichert. Zusätzlich stellt EAS3 noch ein Kommandozeilenprogramm zur Konversion der Daten in andere Datenformate zu Verfügung, sowie einige Standards zur Auswertung und Analyse der Daten.

1.7 Der Gang der Untersuchung

Kapitel 1 – Einleitung Die in diesem Kapitel gelegten Grundsteine werden im weiteren Gang der Untersuchung aufgenommen. Im Verlauf dieses einleitenden Kapitels zeigt sich, dass Forschungsdatenmanagement im HPC vor Herausforderungen steht, seien diese technischer, organisatorischer oder rechtlicher Natur. Dies wurde speziell im Unterkapitel 1.3 als Problemanalyse herausgearbeitet. Darüber hinaus wurde der Forschungsbeitrag geklärt sowie die Methodologie der Arbeit festgelegt.

Kapitel 2 – Verwandte Arbeiten Vorarbeiten und verwandte Arbeiten werden im 2. Kapitel vorgestellt, insofern sie im Laufe der Untersuchung von Bedeutung sind. Einerseits werden Begriffe wie *dunkle Daten* oder *FAIR* eingeführt, andererseits konzeptuelle Arbeiten wie Schichtenmodelle für Forschungsdatenmanagement vorgestellt. Darüber hinaus werden technische Datenmanagementsysteme im HPC Umfeld diskutiert, sofern sie einen Bezug zur eigenen Arbeit besitzen.

Kapitel 3 – Dunkle Daten In Kapitel 3 wird gezeigt, dass die in Kapitel 1 thematisierten Herausforderungen in dunklen, d.h. unsichtbaren und nicht nutzbaren Daten kulminieren, die sich über die Zeit anhäufen. Der Begriff der dunklen Daten wird für das Höchstleistungsrechnen definiert. Er kann analytisch verwendet werden, um unterindzierte oder nicht mehr verfügbare Daten zu kennzeichnen. Dunkle Daten werden im Bandspeichersystem des HLRS nachgewiesen und es wird auf Implikationen rechtlicher, ressourcenökonomischer und ethischer Art hingewiesen. Schließlich wird herausgearbeitet, dass gelungenes Forschungsdatenmanagement die Reduktion dunkler Daten zum Ziel hat.

[22] `https://wiki.iag.uni-stuttgart.de/eas3wiki/index.php/EAS3_File_Format/de`, Zugriff 28.2.2019.

Kapitel 4 – Anforderungen In Kapitel 4 werden aus den Herausforderungen, die in Kapitel 1.3 diskutiert wurden, Anforderungen an ein Forschungsdatenmanagement hergeleitet sowie deren Machbarkeit und Priorität thematisiert. Schließlich werden die Anforderungen in funktionale und nicht-funktionale kategorisiert, auf die sich später die Konzepte beziehen.

Dabei werden die Anforderungen mit Ansprechpartnern aus den o.g. Instituten in teilstrukturierten Interviews expliziert. Außerdem werden diese Anforderungen mit denen aus der bestehenden Literatur abgeglichen, was sich an der jeweiligen Stelle als Zitat wiederfindet. Dies basiert im Wesentlichen auf der Technik des *Requirements Engineering* und somit auf dem Dreischritt *Ermitteln der Herausforderungen – Analyse der Anforderungen – Spezifikation der Anforderungen*. Schließlich wird die Anforderungspezifikation als Liste in Kapitel 4.3 erstellt.

Kapitel 5 – Konzept Das 5. Kapitel stellt meine eigenen konzeptuellen Lösungsansätze für Forschungsdatenmanagement im HPC und deren Implementierung vor. Dazu wird aus den in Kapitel 4 aufgestellten Anforderungen eine Spezifikation von Bausteinen entworfen, um Forschungsdatenmanagement für die o.g. Anwendungsfälle zu etablieren. Diese Bausteine gliedern die Aufgabe des Forschungsdatenmanagements für die computergestützten Ingenieurwissenschaften in überschaubare Einheiten, legen eine Lösungsstruktur fest und definieren eine hierarchische Gliederung. Die Bausteine finden sich in Form von Konzepten wieder, die in eine Schichtenarchitektur für Forschungsdatenmanagement, welche die grundlegende Systemarchitektur festlegt, eingebettet sind. Aus den Anforderungen erfolgt dann die Einführung von *EngMeta*. Dieses stellt ein Metadatenmodell dar, welches in der Lage ist, insbesondere die Spezifika von Forschungsdaten im HPC und in den computergestützten Ingenieurwissenschaften zu erfassen. Dieses dient als Grundlage für die automatisierte Metadatenerfassung, welche in der Nutzungsschicht des Systems nahe am Prozess Metadaten erfasst und schließlich automatisiert in ein Repositorium legen kann. Diese technischen Konzepte werden durch nicht-technische ergänzt: Mit der Rolle des Scientific Data Officers (SDO) kommt ein Ansatz zur Verbesserung von Prozessen ins Spiel. Ebenso können mittels des SDO Fragen der Datenverantwortlichkeit, beispielsweise über dunkle Daten, geklärt werden. Entscheidungskriterien ergänzen seine Tätigkeit.

Diese Konzepte werden als Proof-of-Concept implementiert, wobei im Zentrum das Metadatenmodell *EngMeta* steht. Auf diesem Metadatenmodell aufbauend wird die automatische Metadatenerfassung in einer generischen Form entwickelt, die beide Anwendungsfälle abdeckt und erweiterbar auf andere Formate ist. Schließlich wird ein komplett automatisiert ablaufender Workflow entwickelt, um Forschungsdaten direkt nach ihrer Erzeugung automatisch erfassen zu lassen und die erfassten Metadaten zusammen mit den Forschungsdaten selbst in ein zentrales Repositorium der Universität Stuttgart zu spielen, das im Rahmen des Projekts DIPL-ING aufgesetzt wurde.

Kapitel 6 – Evaluation Die Validierung des Entwurfs aus Kapitel 5 findet in Kapitel 6 statt, wo einige der Konzepte qualitativ evaluiert werden. Hierbei wird das Metadatenmodell, die Metadatenerfassung und die Integration in den Arbeitsprozess evaluiert. Das Metadatenmodell wird hierbei bzgl. Güte und Übertragbarkeit für andere Fälle als die der Thermodynamik und Aerodynamik qualitativ evaluiert. Die Metadatenerfassung wird bzgl. der Anpassbarkeit auf andere Simulationscodes und der Kompatibilität zu typischen Systemumgebungen getestet. Darüber hinaus finden Performance-Tests statt, da die Metadatenerfassung der Anforderung genügen muss, den wissenschaftlichen Arbeitsprozess nicht zu verzögern. Die Integration in den Arbeitsprozess wird schließlich durch den vollautomatisierten Vorgang gezeigt, indem nach einem Simulationslauf direkt Metadaten erfasst und dann Daten und die Metadaten in ein Repositorium gespeichert werden.

Kapitel 7 – Fazit Am Ende der Dissertation steht eine zusammenfassende Diskussion der Ergebnisse und ein Ausblick. Da im Rahmen der Dissertation das Konzept der dunklen Daten auf das Höchstleistungsrechnen bezogen und die eigentliche Definition begrifflich erweitert wurde, gibt es insbesondere in diesem Feld einige Anknüpfungspunkte, die im Ausblick angerissen werden.

Kapitel 2

Verwandte Arbeiten

In diesem Kapitel werden die Vorarbeiten diskutiert, die als Basis für die vorliegende Dissertation von Bedeutung sind. Wurden im einleitenden Kapitel 1 Herausforderungen aufzeigt, werden in diesem Kapitel ähnliche Vorhaben beleuchtet und grundlegende Definitionen getroffen. Es werden zunächst Begriffe eingeführt, die für die Dissertation von Bedeutung sind. Ebenso werden Metadatendefinitionen sowie Definitionen ihrer Eigenschaften und die Darstellung von Qualitätskriterien vorgenommen. Konzeptuelle Arbeiten, wie Schichtenmodelle sowie Ansätze von Forschungsdatenmanagement in angrenzenden Bereichen, werden vorgestellt. Schließlich werden bestehende Datenmanagementsysteme thematisiert, die relevant für das Dissertationsvorhaben sind.

2.1 Dunkle Daten

Im Bereich der dunklen Daten finden sich wenig Vorarbeiten, insbesondere keine, die sich speziell auf den Bereich des Höchstleistungsrechnens oder die computergestützten Ingenieurwissenschaften beziehen. Dunkle Daten werden dabei bisher von Unternehmen oder der Industrie rein phänomenologisch verwendet, um Datenbestände zu kennzeichnen, die vorhanden sind, aber nicht genutzt werden[23]. Hier werden sie im Zusammenhang mit Sensordaten und dem *Internet of Things* thematisiert (Trajanov u. a., 2018) oder vor dem Hintergrund, dass Unternehmen durch die heutige Speichertechnologie in der Lage sind, Daten zunächst ohne konkretes Ziel zu erheben. Abbildung 2.1 zeigt, bezogen auf das Beispiel *Stadt*, die täglich erzeugten Datenmengen und ihren tatsächlichen genutzten Anteil, der in den meisten Bereichen bei 0.1 % liegt. In diesem Sinne werden dunkle Daten also vor allem als Daten, die nicht genutzt werden, bestimmt.

Begrifflich und über dieses Verständnis von dunklen Daten als ungenutzte Daten hinausgehend werden sie bisher nur von zwei Autoren thematisiert: Thomas Goetz (Goetz, 2007) definiert dunkle Daten als Daten aus fehlgeschlagenen Experimenten bzw. als die Daten, die negative Forschungsresultate repräsentieren. Bryan Heidorns Definition (Heidorn, 2008; Heidorn, Stahlman und Steffen, 2018) erfasst die Form der Daten und bezieht sich auf schlecht beschriebene und unverfügbare Daten.

[23] `https://www.gartner.com/it-glossary/dark-data`, Zugriff 19.4.2019.

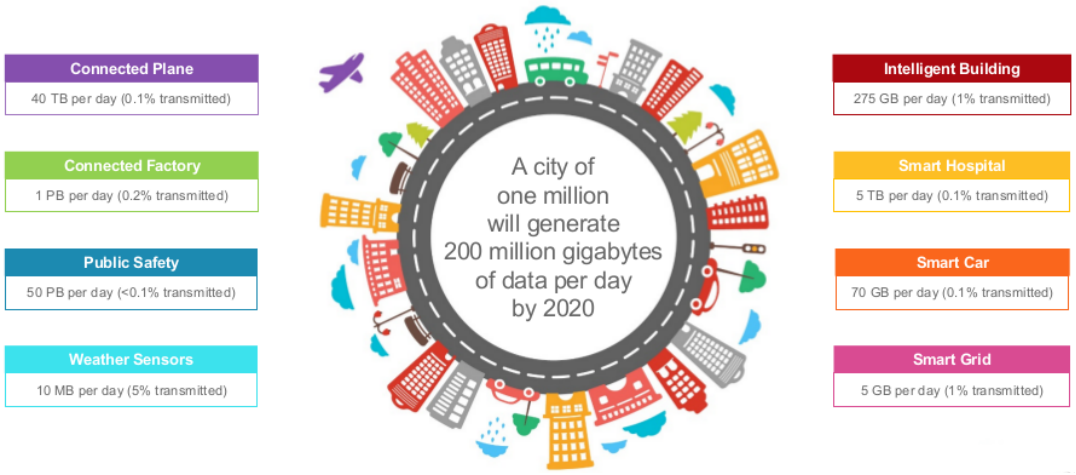

Abbildung 2.1: Täglich erzeugte Datenmenge und davon genutzte (übertragene) Datenmenge bezogen auf das Beispiel *Stadt* (CISCO Systems, 2016).

Meine Arbeit zum Begriff der dunklen Daten knüpft vor allem an die Definitionen von Heidorn an und wurde teilweise in (Schembera und Durán, 2019) vorab veröffentlicht. Da der Begriff der dunklen Daten zentral für das Dissertationsvorhaben ist, wird seiner Explikation, Kritik und Erweiterung auf das Höchstleistungsrechnen ein eigenes Kapitel gewidmet, nämlich Kapitel 3.

2.2 Metadaten

2.2.1 Definition

Metadaten werden gemeinhin als „Daten über Daten“ (Hey und Trefethen, 2003; Rat für Informationsinfrastrukturen, 2016) und als strukturierte Form der Wissensrepräsentation definiert und gelten als wissenschaftlicher Goldstandard, um die Daten nachnutzbar zu machen:

> „Extensive, highly structured metadata often are seen as a holy grail, a magic chalice both necessary and sufficient to render sharing and reusing data seamless, perhaps even automatic.“ (Edwards u. a., 2011, S. 672)

Schon 1994 wird diese Sichtweise in (Bretherton und Singley, 1994) als verkürzt kritisiert, da die Definition von Metadaten stark auf den Kontext ankommt. Für Systemarchitektinnen und -architekten sind Metadaten Informationen auf der Systemebene wie Dateinamen. Für Datenbankarchitektinnen und -architekten sind diese relationale Schemata, für Physikerinnen und Physiker dagegen beispielsweise die Kalibrierungsinformation der Experimente. Allerdings müssten Metadaten als Austauschinformation verstanden werden, in der menschliche Urteilskraft unersetzbar bleibt. Edwards et al. gehen noch einen Schritt weiter und erheben Metadaten zur Kommunikationsform, die kein festes Endprodukt hat und sich im Fluss befindet (Edwards u. a., 2011):

> „We propose an alternative view of metadata, focusing on its role in an ephemeral process of scientific communication, rather than as an enduring outcome or product.“ (Edwards u. a., 2011, S. 667)

Metadaten dienen somit dazu, nicht mit der Erstellerin oder dem Ersteller direkt sprechen zu müssen, sondern sich allein auf die annotierte Information verlassen zu können. Sie nehmen eine Vermittlungsrolle ein und sollen Reibungsverluste („Science Friction“), wie sie bei jeglichem Austausch von Daten durch Misskommunikation stattfinden, eindämmen. Die Aufgabe von Metadaten ist es, ein gemeinsames Verständnis zu erschaffen und somit zur Kommunikationsform zu werden. Metadaten müssen aber um Prozesse ergänzt werden, ohne die sie sinnlos sind:

> „Metadata products can be powerful resources, but very often – perhaps even usually – they work only when metadata processes are also available.“ (Edwards u. a., 2011, S. 668)

Demnach können Metadatenprodukte, also z.B. Metadatenmodelle, nur funktionieren, wenn man diese als nicht-fixe Kommunikationsform begreift, die auch die Prozesse mit einschließt. Die Autorinnen und Autoren erwähnen hier explizit das Phänomen der „Metadata Friction“ (Edwards u. a., 2011, S. 673) und bezeichnen damit den auch in der vorliegenden Dissertation in Kapitel 1.3.2 diskutierten Zusatzaufwand, dem Wissenschaftlerinnen und Wissenschaftler ausgesetzt sind, wenn sie mit Produkten wie Metadatenmodellen arbeiten. Diese Prozesse werden im gemachten Vorschlag (Edwards u. a., 2011) von Metadaten als Kommunikationsform mit gedacht und müssen zu sinnvollem Metadatenmanagement gehören. Die Sichtweise, dass Metadaten „Daten über Daten“ seien, wird auch in (Deelman u. a., 2010) als zu unkonkret kritisiert. (Greenberg, 2003) folgend wird diesem Konzept der Ansatz gegenübergestellt, dass Metadaten strukturierte Daten über ein Objekt sind, die auch Funktionen dieses Objektes unterstützen:

> „structured data about an object that supports functions associated with the designated object“ (Greenberg, 2003, S. 1876)

Metadaten haben also auch hier eine Doppelfunktion: Sie sind eine strukturierte Beschreibung einiger Merkmale, aber sie unterstützen auch den Arbeitsprozess oder die Interpretation dieser Merkmale. Beispielsweise können Metadaten einen Datenpunkt mit dem Fehlerbalken beschreiben, indem sie charakterisieren, um was für eine Art Fehler es sich genau handelt. Somit würde die Interpretation des Datenpunktes durch Metadatenbeschreibung unterstützt werden.

Die generelle Aufgabe von Metadaten ist es, den Objektbezug herzustellen, also das Objekt und seine Eigenschaften oder Teile davon beschreibbar zu machen. Metadaten können dazu unabhängig vom Objekt definiert werden, sie sind nicht notwendigerweise ein Teil des Objekts selbst. Metadaten sind eine Abstraktion von konkreten Objekteigenschaften und konstruiert in dem Sinne, dass sie keine objektinhärenten Eigenschaften sind, sondern diese vom Menschen expliziert werden müssen.

Metadaten bilden eine einheitliche Beschreibungsebene für ein Datenobjekt. Trotz einer einheitlichen Beschreibungsebene sind Metadaten immer noch Interpretationen, die aber nachvollziehbar erstellt und hergeleitet auf eine Basis eines gemeinsamen Verständnisses gestellt werden (Odebrecht, 2018), womit die „Metadata Friction" möglichst reduziert werden soll.

2.2.2 Metadatenmodelle

Metadatenmodelle aufzuzählen und im Einzelnen zu besprechen, würde den Rahmen der Dissertation sprengen[24]. Die im Folgenden näher vorgestellten Metadatenmodelle sind ausschließlich Modelle von hoher Relevanz für das Dissertationsvorhaben. So stammt *EngMeta* aus einer Kombination der vorgestellten Modelle, was in Kapitel 5.2.2 genau erläutert wird. Da die genannten Metadatenmodelle jeweils nur einzelne Aspekte von ingenieurwissenschaftlichen Forschungsdaten (nur Teile von allgemeinen, technischen, prozessspezifischen oder fachspezifischen Eigenschaften) beschreiben und für die Ingenieurwissenschaften nicht spezifisch genug sind, wurde schließlich die Entwicklung von *EngMeta* vorangetrieben.

CERA-2.5

Das Climate and Environmental Retrieval and Archive (CERA) Metadatenmodell ist ein Metadatenmodell zur Beschreibung von klimawissenschaftlichen Daten. Es wurde 1998 vorgestellt (Lautenschlager u. a., 1998) und ist ein Beispiel für ein disziplinspezifisches Metadatenschema. Es liegt mittlerweile in Version 2.5 vor[25] und dient beispielsweise als Grundlage zur Datenbeschreibung für das WDCC[26]. Für einen Datensatz können 42 Blöcke definiert werden, die in die 12 Gruppen METADATA_ENTRY, CAMPAIGN, CONTACT, COVERAGE, DISTRIBUTION, ENTRY_CONNECT, KEY_CONNECT, LANGUAGES, LOCATION_CONNECT, PARAMETER, REFERENCE, SPATIAL_REFERENCE kategorisiert werden, was in Abbildung 2.2 dargestellt ist. Dabei enthält die Gruppe COVERAGE beispielsweise die räumlichen und zeitlichen Informationen über die Daten, die für die Klimawissenschaften von hoher Bedeutung für die wissenschaftliche Arbeit sind. Die Gruppe METADATA_ENTRY bildet die zentrale Entität, in der alle Metadateninformationen der anderen Blöcke zusammen laufen und repräsentiert einen Datensatz. Bei CERA-2.5 handelt es sich somit um ein auf die Klimawissenschaften zugeschnittenes, datenzentrisches Metadatenmodell, welches fachspezifische Informationen, wie die räumlichen und zeitlichen Informationen in der Gruppe COVERAGE, mit allgemeinen deskriptiven und Prozessmetadaten vereinigt, um ein umfassendes Bild von den erzeugten klimawissenschaftlichen Daten zu bekommen.

[24] Unter `http://jennriley.com/metadatamap/` (Zugriff 7.1.2019) findet sich eine eindrucksvolle Visualisierung von 105 Metadatenmodellen- und standards aus dem Bereich der Kulturgüterbeschreibung des Kulturerbes.

[25] `https://www.dkrz.de/up/systems/cera/data_model`, Zugriff 15.2.2019.

[26] WDCC = World Data Centre for Climate, siehe Kapitel 2.4.2.

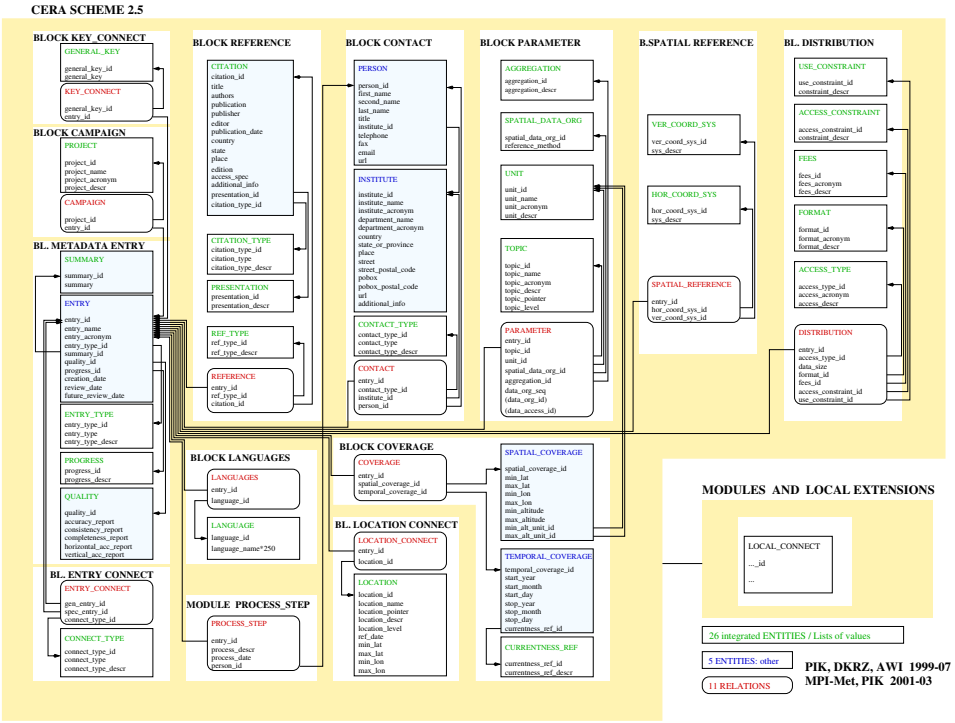

Abbildung 2.2: Das CERA-2.5 Metadatenmodell zur Beschreibung von Daten aus den Klimawissenschaften (Lautenschlager u. a., 1998).

CodeMeta

CodeMeta stellt eine spezialisierte Sprache zur Beschreibung von Software zur Verfügung[27], deren Hauptaufgabe in der Zitierbarmachung von Software jeglicher Art liegt. Dabei modelliert CodeMeta ein Software-Objekt in JSON Linked Data (JSON-LD) Notation. Mittels CodeMeta lassen sich Beschreibungen, wie das genutzte Software-Repository, die Programmiersprache, die Laufzeitumgebung usw. ablegen. Die komplette Auflistung findet sich auf der CodeMeta-Website[28].

CML

Die Chemical Markup Language (CML) ist ein Metadatenschema, welches als XML Schema vorliegt und für Metadatenverwaltung in der Chemie ausgelegt ist (Murray-Rust und Rzepa, 2011). Es wurde ursprünglich für die Darstellung von chemischen Formeln entwickelt und über die Zeit auf angrenzende Bereiche erweitert, insbesondere auch auf den Bereich der computergestützten Chemie bzw. der Molekulardynamiksimulation. Diese Erweiterung wird *CMLComp* genannt und wurde bis 2012 entwickelt[29]. Dabei bildet das Metadatenmodell auch Informationen, wie die Maschinenkonfiguration bzw. Rechenumgebung, Kontrollparameter, die Berechnungsmethode, thermodynamische Größen wie Druck, Temperatur sowie Informationen über

[27] `https://codemeta.github.io/`, Zugriff 7.1.2018.
[28] `https://codemeta.github.io/terms/`, Zugriff 7.1.2018.
[29] `http://homepages.see.leeds.ac.uk/~earawa/CMLComp/index.html`, Zugr. 8.2.2019.

die benutzten Algorithmen ab[30]. CMLComp ist für die computergestützte Chemie konzipiert und ermöglicht einen hohen Detaillierungsgrad. Beispielsweise lassen sich in dieser Sprache auch Moleküle modellieren. CMLComp ist in der Lage, sowohl Eingabe- als auch Ausgabedaten zu repräsentieren (Phadungsukanan u. a., 2012). Für das Anliegen der Dissertation ist es allerdings zu spezifisch auf die computergestützte Chemie zugeschnitten.

DataCite

DataCite ist ein Metadatenmodell zur Beschreibung von Datensätzen in einer persistenten Art und Weise (Ammann u. a., 2011), die eng mit dem Konzept der DOI[31] verknüpft ist. DataCite ist gleichzeitig der zentrale Dienstanbieter zur Vergabe von DOIs. Die Entwicklung von DataCite wird seit 2009 von einer Arbeitsgruppe vorangetrieben und neue Versionen werden stetig veröffentlicht. Die Metadatenfelder gliedern sich in die drei Kategorien *mandatory* (verbindlich), *recommended* (empfohlen) und *optional*, wobei die verbindlichen Felder in Version 4.1[32] sechs Attribute umfassen, nämlich *Identifier*, *Creator*, *Title*, *Publisher* und *PublicationYear* sowie *ResourceType*. Dies sind die typischen Zitationsmetadaten, die Daten zitierbar machen sollen. Die verbindlichen Attribute entsprechen den Zitationsmetadaten und spalten sich noch einmal in weitere Teile auf. So besteht z.B. der Identifier aus dem String selbst sowie dem Typ des Identifiers, dessen Wert z.B. *DOI* annehmen kann. Zusätzlich zu den verbindlichen Attributen stehen 13 weitere empfohlene oder optionale Attribute. Neben dem Metadatenmodell an sich umfasst DataCite die Infrastruktur mit einem Registrierungsdienst und einem Metadatenspeicher, der die Metadaten aller registrierten Objekte hält. Hierzu müssen die Metadateninformationen im XML-Format mittels eines Browsers oder einer API hochgeladen werden, womit das Objekt eine DOI bekommt und als registriert gilt (Neumann und Brase, 2014). Datenzitierung und Metadatenbeschreibung sollen mit DataCite in Einem gelöst werden.

Dublin Core

Dublin Core ist ein bibliographisches Format zur Beschreibung von Dokumenten im Internet, welches ab 1994 von der Dublin Core Metadata Initiative (DCMI) entwickelt wurde[33]. Dabei enthält es 15 Kernfelder (*core elements*), deren Nutzung von der DCMI empfohlen wird. Dabei gilt Dublin Core als Quasistandard bei der Beschreibung von Dokumenten oder auch Kulturgütern in Museen. Ebenso nutzen viele Repositorien für die Dokumentenverwaltung Dublin Core, wie z.B. Fedora Commons.

[30] `http://www.xml-cml.org/convention/compchem`, Zugriff 8.2.2019.

[31] DOI steht für *Digital Object Identifier* (dt. *Digitaler Objektbezeichner*) und ist ein eindeutiger, persistenter Bezeichner für digitale Objekte. Die Struktur und Funktionsweise sind durch den ISO-Standard ISO 26324 festgelegt (*Information and documentation – Digital object identifier system* 2012).

[32] `https://schema.datacite.org/meta/kernel-4.1/doc/DataCite-MetadataKernel_v4.1.pdf`, Zugriff 4.1.2019.

[33] `http://dublincore.org/`, Zugriff 7.1.2019.

ExptML

Die Experiment Markup Language (ExptML) ist ein spezialisiertes Metadatenmodell zur Beschreibung des wissenschaftlichen Arbeitsprozesses im Hinblick auf Experimente[34]. Die Beschreibungssprache ist als XML Schema (XSD) definiert und besteht aus 26 modularen Datentypen zur Beschreibung von wissenschaftlicher Arbeit, wobei das Metadatenmodell UnitsML zur Beschreibung von Einheiten herangezogen wird[35]. Mit ExptML sollen insbesondere digitale Daten aus Experimenten, aber auch aus Simulationen besser in Laborbücher integriert werden können. Dabei ist ExptML stark am wissenschaftlichen Arbeitsprozess orientiert, welcher von einer Literaturrecherche über die Kommunikation mit anderen Wissenschaftlerinnen und Wissenschaftlern über das Experiment selbst und die Kommunikation der Resultate bis zur Veröffentlichung eines Reports oder Papers geht. Beispielsweise gibt es für die Erfassung der Kommunikation mit anderen, sei es direkt durch Projekttreffen oder über E-Mails, einen Datentyp *Communication*, welcher den Kommunikationsvorgang erfasst. Als zentrale Komponente gibt es den Datentyp *Instrument*[36], welcher die Beschreibung des Experimentierinstruments umfasst.

PREMIS

Das Metadatenmodell PREMIS (PREMIS Data Dictionary for Preservation Metadata) basiert auf der konzeptuellen Arbeit von OAIS[37] und hält an der prinzipiellen Aufteilung von Metadaten in deskriptive, administrative, strukturelle und Preservations-Metadaten fest. Dabei stellt es einen Standard für Erhaltungsmetadaten (Preservation) dar, d.h. es berücksichtigt alle Belange, um Daten über lange Zeiträume hinweg verfügbar zu halten (Caplan, 2009). Der Hauptnutzen von PREMIS liegt dabei neben dem Repositoriumsgedanken darauf, archivierte Datenobjekte austauschbar zu machen, was durch die Preservationsinformationen erhalten werden soll. Dabei sollten diejenigen, die ein Repositorium planen, PREMIS als Kontrollliste verwenden und potentielle Repositoriumssoftware auf PREMIS-Nutzung prüfen, um optimale Datenerhaltung zu gewährleisten. Dazu definiert der PREMIS-Standard keine Metadatenelemente im eigentlichen Sinne, sondern semantische Einheiten. Diese unterscheiden sich in ihrer Abstraktheit: Während ein Metadatenelement eine definierte Art darstellt, Informationen in einem Schema, in einer Datenbank, etc. zu repräsentieren, gibt eine semantische Einheit keinerlei Aufschluss über die konkrete Repräsentation, sondern ist möglichst allgemein gehalten und definiert nur die Semantik. Das Datenmodell von PREMIS gliedert sich dann in die logischen Entitäten *Object* (*Objekt*), *Agent* (*Agent*), *Event* (*Ereignis*) und *Rights* (*Rechte*). Jede dieser Entitäten,

[34] `http://exptml.sourceforge.net/`, Zugriff 7.1.2019.

[35] Das vom amerikanischen *National Institute of Standards and Technology* (NIST, deutsch: Nationales Institut für Standards und Technologie) gepflegte Schema UnitsML (`http://wiki.eclass.eu/wiki/UnitsML`, Zugriff 29.1.2019.), ist ein spezielles Metadatenschema zur Beschreibung von wissenschaftlichen Maßeinheiten.

[36] `http://exptml.sourceforge.net/instrument`, Zugriff 7.1.2019.

[37] Siehe Kapitel 2.3.2.

bei denen das Objekt zentral ist, wird ihrerseits mit entsprechenden semantischen Einheiten beschrieben, wobei das Objekt das jeweilig zu verwaltende Datenobjekt ist. Die semantischen Einheiten, die das Objekt beschreiben, sind ein eindeutiger Identifier, Unveränderlichkeitsinformationen wie die Checksumme (mit der Information über den Algorithmus). Darüber hinaus sind es die Größe und das Format des Datenobjekts, dessen Name, Informationen über die Erzeugung, wie die Datenobjekte blockiert werden können, Speicherort, digitale Signatur und die Verbindungen zu anderen Objekten. Die genaue Auflistung der semantischen Einheiten findet sich in (Caplan, 2009) bzw. für den aktuellen Standard 3.0 auf der PREMIS-Website[38].

PROV/ProvOne

Das PROV Metadatenmodell[39] legt den Fokus auf Prozessmetadaten bzw. die Datenprovenienz im Allgemeinen und ist eine Empfehlung der W3C[40]. Es kann mittels des Modells beschrieben werden, wie Daten oder Objekte grundsätzlich erstellt, generiert oder übergeben wurden. Dabei steht eine *Activity* im Zentrum, die beschreibt, wie eine Entität generiert wurde und repräsentiert somit einen Prozessschritt, der zum Datenobjekt o.Ä. geführt hat. Eine *Entity* repräsentiert ein physikalisches, digitales, konzeptuelles oder sonstiges Objekt und wird durch verschiedene Attribute beschrieben wird. Ein *Agent* verantwortet eine Activity, die z.B. zu einem Datenobjekt geführt hat und kann bspw. eine Person oder auch Software sein. Das komplette Datenmodell liegt als XML-Schema[41] vor. Die Erweiterung ProvONE[42] konkretisiert das Metadatenmodell zur Dokumentation wissenschaftlicher Arbeitsprozesse. Dabei deckt ProvONE die Teile *Workflow*, *Trace* und *Data Structure* ab, wobei *Trace* als Activity gesehen werden kann und die Entstehungsgeschichte abbildet.

2.2.3 Anwendungsprofile

Die Begrifflichkeit „Anwendungsprofil“ bezeichnet die Anpassung eines oder mehrerer Metadatenmodelle für einen spezifischen Anwendungsfall.

> „We define application profiles as schemas which consist of data elements drawn from one or more namespaces, combined together by implementors, and optimised for a particular local application.“ (Heery und Patel, 2000, S. 1)

Anwendungsprofile erweitern bestehende Metadatenmodelle in einer pragmatischen Art und Weise im Hinblick auf eine gewisse Anwendung, indem sie diese zusammenführen und für einen spezifischen Anwendungsfall nutzbar machen. Darüber

[38] `https://www.loc.gov/standards/premis/`, Zugriff 26.12.2018.
[39] `https://www.w3.org/TR/2013/NOTE-prov-overview-20130430/`, Zugriff 7.1.2019.
[40] W3C ist das *World Wide Web Consortium (kurz W3C)*. Dieses stellt das Organ zur Technikstandarisierung im WWW dar.
[41] `https://www.w3.org/TR/2013/NOTE-prov-xml-20130430/`, Zugriff 7.1.2019.
[42] `http://vcvcomputing.com/provone/provone.html`, Zugriff 7.1.2019.

hinaus muss das Anwendungsprofil dokumentiert werden. Dabei hat es folgende Eigenschaften. Die Nutzung von Namensräumen der ursprünglichen Metadatenmodelle ist dabei zur eindeutigen Identifikation von Metadatenelementen obligatorisch. Werden neue Metadatenelemente eingeführt, muss ein eigener Namensraum definiert werden, der auch gepflegt werden muss. Damit gilt die Arbeit nicht mehr als Anwendungsprofil, sondern als eigenes Schema. Die Wertebereiche von Standardschemata können in einem Anwendungsprofil angepasst werden. Diese Definition wird vom *Singapore Framework for Dublin Core Application Profiles*[43] aufgenommen, um die Brücke zu schlagen zwischen fachspezifischen Anforderungen und der Anforderung nach Interoperabilität.

2.2.4 Metadatenerfassung

Jane Greenberg unterscheidet in (Greenberg, 2004) zwei Arten der Erfassung, nämlich *Extraktion* und *Harvesting*. Metadatenextraktion ist die (teil-)automatisierte Erfassung aus den ursprünglichen Ressourcen zur Erstellung von strukturierten Metadaten. Harvesting ist dagegen die Technik, bereits erfasste vorliegende strukturierte Metadaten z.B. in einem Repository zu zentralisieren oder auszutauschen.

In (Park und Brenza, 2015) werden 39 Metadatenerfassungstools vorgestellt und bewertet. Die Studie kommt zu der Einschätzung, dass die meisten Tools nur wenige Metadatenelemente erfassen können und sich nicht ohne größere Anstrengungen in den Arbeitsprozess integrieren lassen. Dabei ist es die zentrale Aufgabe für die Integration, die Metadatenerfassung so zu gestalten, dass sie sich nahtlos in den wissenschaftlichen Prozess einfügt. Dem Artikel zufolge ist dafür finanzielle Unterstützung, geschultes Personal sowie Training unabdingbar.

In der Dissertation von Ajinkya Prabhune (Prabhune, 2018) wird ein generisches Konzept zum Metadatenmanagement vorgeschlagen, welches auch im MASi–Repositorium zum Einsatz kommt[44]. Dabei gibt es eine Komponente zur Metadatenerfassung, die auf dem, auch in (Park und Brenza, 2015) mehrfach erwähnten Framework, Apache Tika[45] basiert. Apache Tika bietet Extraktionsmöglichkeiten für viele Standard-Dateiformate. Allerdings muss für fachspezifische Daten und Metadatenformate einiges an Implementierungsaufwand investiert werden: Der Extraktor muss in Java von den jeweiligen Wissenschaftlerinnen und Wissenschaftlern für den jeweiligen Simulationscode und das Metadatenmodell implementiert werden.

[43] `http://dublincore.org/documents/singapore-framework/`, Zugriff 17.3.2019.

[44] MASi (Metadata Management for Applied Sciences) ist ein Dienst für Forschungsdatenmanagement, welcher sich in einer Pilotphase befindet und dabei nicht disziplinspezifisch ist (Grunzke u. a., 2019). Dabei nutzt es als technologische Basis den KIT Data Manager (`http://datamanager.kit.edu/index.php/kit-data-manager`, Zugriff 1.3.2019).

[45] `https://tika.apache.org/`, Zugriff 2.3.2019.

2.2.5 Metadatenqualität und -evaluation

Metadatenqualität zu definieren und erfassen, ist ein schwieriges Unterfangen:

> „Like pornography, metadata quality is difficult to define. We know it when we see it, but conveying the full bundle of assumptions and experience that allow us to identify it is a different matter." (Bruce und Hillmann, 2004, S. 1)

Metadatenqualität kann über die Probleme bei Abwesenheit von Metadaten definiert werden, wobei dies nicht ausreicht:

> „[...] we too have found that it is difficult to talk about quality without also talking about things that betray its absence, but we believe that trying to comprehend quality by enumerating defects risks sacrificing an organized view [...]. Instead, we attempt a systematic, domain- and method-independent discussion of quality indicators." (Bruce und Hillmann, 2004, ebd.)

Metadatenqualität wird vor allem im Bereich der Bibliotheks- und Informationswissenschaften diskutiert, die in (Bruce und Hillmann, 2004) einen systematischen, methodenunabhängingen Ansatz empfehlen, der über Metadaten als Suchkriterien und zum Unterstützen von Suchanfragen hinausgeht. Dabei schlagen die Autorinnen und Autoren sieben Kriterien zur Definition von Metadatenqualität vor:

1. **Vollständigkeit**: Das beschriebene Objekt soll so vollständig wie möglich beschrieben werden. Darüber hinaus sollten alle Ausprägungen so viele Elemente des Modells wie möglich nutzen.

   Zur Überprüfung, ob der Metadatensatz das Objekt komplett beschreibt und ob alle Elemente für jedes Objekt genutzt werden, können das Anwendungsprofil, die Dokumentation und über Beispieldatensätze geprüft werden.

2. **Genauigkeit**: Mehrdeutigkeiten sollten vermieden werden, zumindest muss die gespeicherte Information korrekt sein.

   Es wird geprüft, ob akzeptierte Methoden zur Extraktion verwendet wurden und wie valide Werte und Struktur sichergestellt werden. Darüber hinaus sollte geprüft werden, wie Standardwerte garantiert werden. All das lässt sich über Beispieldaten oder die Dokumentation feststellen.

3. **Herkunft**: Es sollten auch Informationen über den Entstehungsprozess der Metadaten verfügbar sein.

   Dies umfasst nicht nur die Überprüfung, wer die Metadaten extrahiert, erstellt oder transformiert hat, sondern auch wie sie erstellt oder extrahiert wurden und durch welche Transformationsprozesse sie gegangen sind. Dies kann sich durch Informationen in den Metadaten feststellen lassen.

4. **Übereinstimmung mit den Erwartungen**: Der Punkt betrifft insbesondere die Einbeziehung der Forschungsgemeinschaft, für die das Metadatenmodell entworfen wird. Ziel ist es, Missverständnisse zu vermeiden. Hier können auch kontrollierte Vokabulare helfen.

   Zur Überprüfung muss analysiert werden, ob die Metadaten das beschreiben, was sie vorgeben und die genutzten kontrollierten Variablen mit dem Verständnis des Objektes einhergehen. Die Erwartungen der Community müssen ebenso geprüft werden. Dies kann nur durch die Dokumentation sichergestellt werden, über Beispieldaten oder die Rückkopplung mit der Forschungscommunity.

5. **Logische Konsistenz und Kohärenz**: Sicherstellung, dass Elemente so definiert werden, dass sie Standards entsprechen oder Standardmethoden beim Metadatenhandling angewandt werden.

   Hierbei muss überprüft werden, ob die Daten in den Elementen konsistent sind und wie sie im Vergleich zu anderen Daten in der Fachcommunity dastehen. Dies kann nur durch Dokumentation oder manuelle Überprüfung geschehen.

6. **Aktualität**: Aktualität kann einerseits durch persistente IDs garantiert werden, die allerdings auch synchron gehalten werden müssen. Die Synchronisation kann aber bei der Erstbeschreibung oft nicht aufrecht erhalten werden (wenn das Datenobjekt schon veröffentlicht wurde, das Metadatentagging aber dauert, und es nachgezogen werden muss). Das gilt es zu vermeiden.

   Es muss analysiert werden, ob die Metadaten sowie die kontrollierten Vokabulare bei Ressourcenveränderung auch mit aktualisiert werden, was über Tests ermittelt werden kann.

7. **Zugreifbarkeit**: Grenzen (technische, organisatorische, aber vor allem ökonomische und handelsbezogene), die die Zugreifbarkeit der Metadaten einschränken, müssen abgebaut werden.

   Hierbei muss überprüft werden, ob angemessene Elemente für die Nutzerinnen und Nutzer vorliegen und das Modell bezahlbar genutzt werden kann und Erweiterungen erlaubt. Dies kann nur durch die Dokumentation, Erfahrung und die Art des Schemas festgelegt werden.

Daneben bietet das *Framework of Guidance for Building Good Digital Collections*[46] der *National Information Standards Organization* ähnlich wie der vorherige Ansatz sechs Prinzipien, die Metadatenqualität definieren, nämlich:

1. „**Metadata Principle 1**: Good metadata conforms to community standards in a way that is appropriate to the materials in the collection, users of the collection, and current and potential future uses of the collection.“ (NISO, 2007, S. 63)

[46] `http://framework.niso.org/`, Zugriff 13.3.2019

Dieses Prinzip bedeutet, dass Metadaten einem anerkannten Standard entsprechen oder zumindest (in Teilen) auf diesen verweisen sollen. Proprietäre oder selbst gebaute Schemata müssen vermieden werden. Das Design des Metadatenmodells sollte mittels einer Anforderungsanalyse geschehen und auch Definitionen sowie bewährte Vorgehensweisen enthalten.

2. „**Metadata Principle 2**: Good metadata supports interoperability.“ (NISO, 2007, S. 76)

   Damit das zweite Prinzip gewahrt wird, sollten gemäß den weiteren Ausführungen im zitierten Text, Metadatenmodelle nur explizite und keine impliziten Informationen enthalten – Metadaten müssen also kohärent, aussagekräftig und global verständlich sein. Dabei müssen Metadaten einfach zwischen Systemen austauschbar und diese in einer zentralen Stelle registriert sein.

3. „**Metadata Principle 3**: Good metadata uses authority control and content standards to describe objects and collocate related objects.“ (NISO, 2007, S. 79)

   Diese Forderung bedeutet, dass Metadaten – sofern möglich – auf Normdateien und kontrollierten Vokabularen basieren sollten, um Mehrdeutigkeiten zu vermeiden.

4. „**Metadata Principle 4**: Good metadata includes a clear statement of the conditions and terms of use for the digital object.“ (NISO, 2007, S. 81)

   Die vierte Forderung besagt, dass die Metadaten eindeutige Aussagen zu Nutzungsrechten und sonstigen lizenzrechtlichen Belangen der beschriebenen Daten enthalten.

5. „**Metadata Principle 5**: Good metadata supports the long-term curation and preservation of objects in collections.“ (NISO, 2007, S. 83)

   Die fünfte Forderung stellt sicher, dass durch administrative Teile in den Metadaten (in diesem Zusammenhang sind technische und Erhaltungs-Metadaten gemeint) die Langzeitarchivierung und Kuration der Daten sichergestellt werden kann.

6. „**Metadata Principle 6**: Good metadata records are objects themselves and therefore should have the qualities of good objects, including authority, authenticity, archivability, persistence, and unique identification.“ (NISO, 2007, S. 85)

   Hier wird zur Forderung erhoben, dass die Metadaten selbst mit Informationen versehen sind, also „Meta-Metadaten“ enthalten, damit sie für Dritte nachvollziehbar bleiben.

Vor allem (NISO, 2007) eignet sich für die Bewertung von Metadatenmodellen, da in diesem die Prinzipien für gute Metadaten allgemein formuliert werden und die Modellierung betrachtet wird. Der Ansatz in (Bruce und Hillmann, 2004) unterscheidet

nicht klar Qualitätsprinzipien für das Metadatenmodell selbst und für die Metadatenausprägungen. Zur statistischen Untersuchung der Qualität von Metadatensätzen wird in (Gavrilis u. a., 2015) das *Metadata Quality Evaluation Model* vorgeschlagen, welches quantitative Metriken für die Kriterien *Completeness*, *Accuracy*, *Consistency*, *Appropriateness*, *Auditability* definiert. Damit können Bestände von Metadaten analysiert und Aussagen über das zugrundeliegende Metadatenmodell getroffen werden. Einen Überblick über quantitative Ansätze findet sich in (Tani, Candela und Castelli, 2013), wobei festgestellt wird, dass es keinen Konsens im Verständnis gibt, außer in der akzeptierten Sichtweise, dass Metadatenqualität subjektiv, kontextabhängig und ein mehrdimensionales Problem ist. Durch Big Data würden die Probleme noch weiter verstärkt, da die Heterogenität zunähme.

2.3 Konzeptuelle Arbeiten

2.3.1 FAIR-Prinzipien

Die FAIR-Prinzipien stellen Richtlinien dar, um die Wiederverwendbarkeit von Forschungsdaten sicherzustellen. Sie wurden seit 2014 in einer Gruppe um Mark D. Wilkinson in einer Reihe von Workshops erarbeitet und in einem paradigmatischen wissenschaftlichen Beitrag in *Nature* vorgestellt (Wilkinson u. a., 2016). Die FAIR-Prinzipien haben dabei im Unterschied zu anderen Initiativen zur Verbesserung des Forschungsdatenmanagements, neben der Unterstützung der menschlichen Nachnutzung, vor allem die Verbesserung der automatisierten Auffindbarkeit und Nutzung der Daten zum Ziel.

Die Autorinnen und Autoren stellen zunächst fest, dass *gutes* Datenmanagement zwar erwünscht, aber nirgends klar definiert sei, was unter *gutem* Datenmanagement zu verstehen sei, wie es ausgestaltet sein müsse und dass eine begriffliche Klärung von hohem Interesse für alle beteiligten Parteien wäre. Die begriffliche Klärung wird im Artikel vorgenommen und basiert auf vier Prinzipien, nämlich den Prinzipien der Findbarkeit (Findability), Zugreifbarkeit (Accessability), Interoperabilität (Interoperability) und Nachnutzbarkeit (Re-usability) von Daten. Bei den FAIR-Prinzipien handelt es sich den Autorinnen und Autoren zufolge um Idealvorstellungen, die als Handlungsanweisungen gelten sollen und keine harten Standards darstellen. Die vier Prinzipien stehen zwar miteinander in Verbindung, können aber auch für sich alleine stehen oder beliebig kombiniert werden. Sie liefern zunächst domänenübergreifende, technologieunabhängige und übergeordnete Richtlinen, die für den jeweiligen Fachbereich oder Anwendungsfall ausformuliert, konkretisiert und dann technisch umgesetzt werden müssen. Die vier Prinzipien sind nur sehr kursorisch in jeweils drei bis vier Sätzen dargestellt und werden im Folgenden kurz genannt. Dabei bezieht sich die Bezeichnung in der Klammerung hinter den Forderungen auf die in der originalen Publikation verwendeten Kürzel, beispielsweise steht *F1* für das erste Prinzip bzgl. Findbarkeit.

Findbarkeit Die Findbarkeit der Daten kann sichergestellt werden, indem Metadaten und Daten mit einer global eindeutigen und persistenten Kennung (Persistent Identifier, PID) versehen sind (F1). Außerdem müssen die Daten mit reichhaltigen Metadaten beschrieben sein (F2) und diese Metadaten wiederum die PID enthalten (F3). Schließlich müssen Daten und Metadaten in einer suchbaren Ressource registriert und indiziert sein (F4).

Zugreifbarkeit Zugreifbarkeit heißt, dass Daten und Metadaten mittels Standardkommunikationsprotokollen und anhand ihres PID erreichbar sein müssen (A1). Dabei muss das Protokoll offen, frei und universell implementierbar sein (A1.1), sowie eine Methode zur Authentifikation und Autorisierung bereitstellen (A1.2). Ebenso müssen die Metadaten bei Unverfügbarkeit der Daten noch zugreifbar bleiben (A2).

Interoperabilität Interoperabilität wird gewährleistet, indem für Daten und Metadaten formale, verfügbare, gemeinsam genutzte sowie breit anwendbare Sprachen zur Wissensrepräsentation verwendet werden (I1). Außerdem müssen Daten und Metadaten Vokabulare nutzen, die den FAIR-Prinzipien entsprechen (I2). Schließlich sollen Daten und Metadaten geeignete Referenzen zu anderen Daten und Metadaten enthalten (I3).

Nachnutzbarkeit Die Nachnutzbarkeit der Daten soll erreicht werden, indem die Metadaten und Daten ausführlich beschrieben sind, und zwar mit einer Vielzahl von präzisen und relevanten Attributen (R1). Dabei müssen sowohl Daten als auch Metadaten mit einer klaren und verfügbaren Datenlizenz ausgestattet werden (R1.1), ihr Ursprung und ihre Herkunft muss detailliert festgehalten (R1.2) und domänenspezifische Standards müssen berücksichtigt werden (R1.3).

Aufgrund ihres allgemeingültigen und nicht auf einzelne Disziplinen zugeschnittenen Charakters haben die FAIR-Prinzipien seit ihrer Veröffentlichung eine große Resonanz erfahren. Zum einen sind es Praktikerinnen und Praktiker, die anhand dieser Richtlinien ein wesentlich zielgerichteteres Forschungsdatenmanagement vorantreiben können. So spiegeln sich auch in der vorliegenden Forschungsarbeit die Prinzipien wider, zum Beispiel finden sie sich im Design des Metadatenmodells *EngMeta* in Kapitel 5.2, aber auch in der Diskussion um Metriken für dunkle Daten in Kapitel 3.1.1. Auf der anderen Seite nehmen immer mehr wissenschaftliche Beiträge oder konkrete Handlungsanweisungen darauf Bezug, z.B. der EU-Report und Aktionsplan der Europäischen Kommission (EU, 2018). Diese sieht die vier Prinzipien so eng miteinander verknüpft, dass sie nur gemeinsam zum Ziel gemacht werden können und schlägt darum einen ganzheitlichen Ansatz vor. Außerdem attestiert der Report nicht nur einen technischen, sondern auch ein kulturellen Handlungsbedarf im Forschungsdatenmanagement. Der Report liefert 27 elaborierte Empfehlungen für

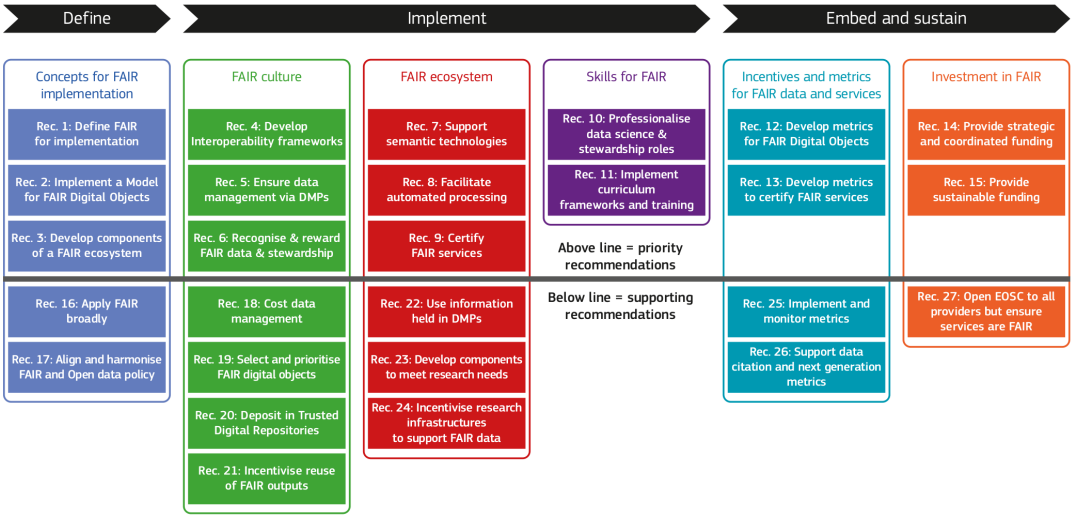

Abbildung 2.3: Konkrete Ausformulierung der FAIR-Prinzipien der EU/EC (EU, 2018), die in den drei Phasen „Definition“, „Implementierung“ und „Nachhaltige Einbettung“ stattfindet und 27 Empfehlungen umfasst.

die verschiedenen Interessengruppen, die in drei Phasen – Definition, Implementierung und nachhaltiger Einbettung – ablaufen, wobei 15 der Empfehlungen priorisiert und die restlichen 12 unterstützende Empfehlungen sind. Die Empfehlungen werden in die folgende Bereiche gruppiert: Konzepte, Kultur, Umgebung, Fähigkeiten, Anreize und Metriken sowie Investitionen. Diese Empfehlungen sind in Grafik 2.3 zusammengefasst und werden für die Zukunft maßgeblich die Forschungslandschaft innerhalb der EU mitprägen. Die in (EU, 2018) vorgeschlagene Erweiterung ist von Bedeutung, da sich die originale Definition zu sehr auf Optimierung der Metadaten stützt und die angrenzenden Bereiche, wie Datenverantwortung usw., nicht betrifft.

Für Forschungsdatenmanagement im HPC haben die FAIR-Prinzipien weitreichende Implikationen. Forschungsdatenmanagement, wie es heute im HPC existiert, erfüllt die meisten in (EU, 2018) genannten und in Abbildung 2.3 empfohlenen Handlungsanweisungen bzgl. der FAIR-Prinzipien nicht, was in den Kapiteln 1.3.1 und 1.3.2 gezeigt wurde. Metadatenmodelle und Datenmanagementpläne fehlen ebenso wie Anreize, die Daten *FAIR* zu halten. Professionelle Datenmanagementrollen und -verantwortlichkeiten sind nicht vorhanden, ebenso fehlt die Integration ins Curriculum. Nachhaltige Finanzierung und Metriken fehlen gänzlich im HPC. Die Umsetzung der Handlungsempfehlungen bzgl. Metadatenmodellen und Datenverantwortlichkeiten ist das Hauptanliegen der Dissertation.

2.3.2 Open Archival Information System

Disziplinübergreifend bietet das Open Archival Information System (OAIS) (OAIS, 2012) eine Anleitung zum Entwurf von Systemen zum Datenmanagement. OAIS ist ein Referenzmodell und ISO-Standard für Archiv-Informationssysteme. Es definiert

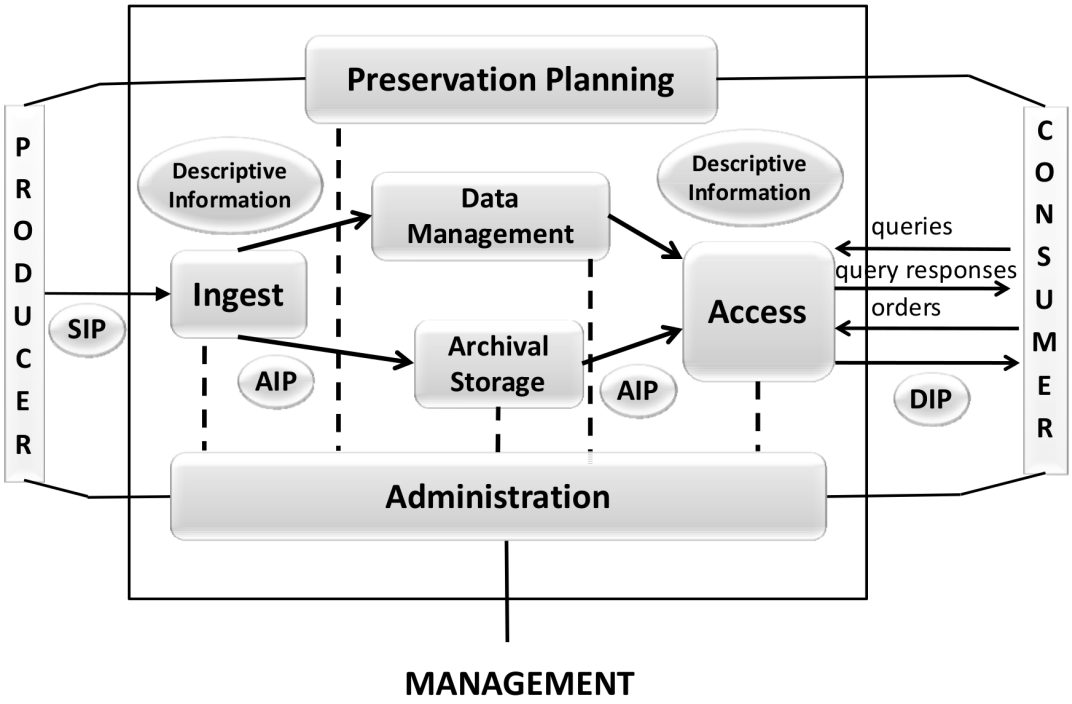

Abbildung 2.4: Die wesentlichen Elemente von OAIS, sowie der Datenfluss und die am System beteiligten Interessengruppen (OAIS, 2012).

das Zusammenwirken von Menschen und technischen Systemen zur Datenarchivierung, wobei es keine konkrete Implementierung, sondern Richtlinien und einen Standard vorgibt. Es entstammt dem Consultative Committee for Space Data Systems (CCSDS) und ist mittlerweile in Version 2 als ISO-Standard 14721:2012 veröffentlicht. Dabei ist OAIS viel weniger paradigmatisch als FAIR, sondern bezieht sich auf das Design von Archivsystemen.

Das OAIS Referenzmodell ist in Abbildung 2.4 dargestellt. Hiernach soll ein OAIS-konformes System Komponenten zum Einstellen (Ingest) von und zum Zugriff (Access) auf Daten bereitstellen, außerdem Archivspeicher (Archival Storage) und Datenmanagement zum Halten der Daten. Flankiert werden sollen diese Komponenten durch die Administration (technisch wie organisatorisch), sowie einer Datenerhaltungsplanung (Preservation Planning), welche z.B. den technologischen Fortschritt bzgl. Bandspeicher mit einbeziehen muss. Eingehende Daten werden als SIP (submission information package) bezeichnet und im Eingabeprozess in AIP (archival information package) umgewandelt, die dann im Archivspeicher abgelegt werden. Wird aus Daten im System zugegriffen, werden DIP (dissemination information package) erstellt und versandt. Darüber hinaus wird das System von den Rollen des Datenproduzenten bzw. der Datenproduzentin, des Datennutzers bzw. der Datennutzerin sowie des Managements bestimmt. Im Spezifikationsdokument werden die einzelnen Entitäten noch weiter aufgelöst. Zusätzlich zur Vorgabe des Systemdesigns wird ein Metadatenmodell als zentrale Komponente definiert, um Daten verwalten zu können. Diesem Metadatenmodell folgend soll ein AIP neben dem Datenobjekt selbst

	(Aschenbrenner et al., 2011)	(Razum, 2011)	(Lautenschlager, 2011)
3	Nutzungsschicht	Dienstschicht	Semantische Ebene
2	Objektverwaltung	Kern-/Dienstschicht	Syntaktische Ebene
1	Storageschicht	Persistenzschicht	Maschinenebene

Tabelle 2.1: Schichtenmodelle von Forschungsdatenmanagementsystemen nach der Literatur.

strukturelle, administrative sowie deskriptive Metadaten enthalten. Strukturelle Metadaten betreffen die Repräsentationsinformationen wie Dateiformatsinformationen, die die Datei überhaupt technisch lesbar machen. Deskriptive Metadaten beschreiben die Daten von einem höheren Standpunkt aus. Administrative Metadaten betreffen vor allem die Datenerhaltung und umfassen Referenz-, Herkunfts-, Kontext- sowie Informationen über die Unveränderbarkeit und die Zugriffsrechte der Daten. Zwar werden im OAIS-Systementwurf immer Daten mit Metadaten zusammen als Paket im Archivspeicher gespeichert, allerdings wird in der Datenmanagementkompomente die deskriptive Information mitgeführt, so dass das Archiv durchsuchbar ist.

Da OAIS ein Referenzmodell darstellt, kann es als Orientierungsrahmen dienen. Die Zerlegung des Gesamtsystems in funktionale Einheiten entspricht nicht dem typischen Schichtenmodell, wie es im Software Engineering verwendet wird. Dieses muss jeweils für den konkreten Nutzungsfall erstellt werden, so dass die volle Kompatibilität nicht notwendig ist. Als erstes Konzept wird auch vorgeschlagen, Daten- und Metadatenspeicherung zu trennen, was speziell im HPC von hoher Priorität ist. Aufgrund der enormen Größe der Daten ist es im HPC nicht möglich, ein effizientes Datenmanagementsystem zu gestalten und beim Durchsuchen jedes einzelne Datenobjekt zu lesen, was z.B. zeitintensive Rückholvorgänge von Bandspeichermedien nötig machen würde. Allerdings ist die Trennung von Daten und Metadaten nicht konsequent zu Ende gedacht, da es sich bei den durchsuchbaren Metadaten in der Datenmanagementkomponente nur um die deskriptiven Informationen handelt.

2.3.3 Schichtenmodelle für Forschungsdatenmanagement

In der Literatur werden drei für ein Forschungsdatenmanagementsystem wesentliche Schichten identifiziert, die je nach Autor verschieden benannt und in Tabelle 2.1 dargestellt sind, sich aber in ihrem Kern ähneln.

Schicht 1 ist die grundlegende Schicht, die als technische Basis für alle weiteren Schichten dient. Auf dieser logischen Ebene wird sichergestellt, dass die Daten unversehrt bleiben, wobei man vor allem auf Technologien wie RAID, die Aufteilung auf zwei oder mehrere Kopien, Checksummen und Erneuerung von Medien nach einer gewissen Frist setzt und dies unter Bit(stream)-Preservation fasst. Auf dieser Schicht werden die Daten physikalisch auf haltbaren, persistenten Datenträgern gespeichert. Auf der Ebene des Datenmanagements (2. Schicht in Tabelle 2.1) findet die Verwaltung von Datenobjekten statt, die durch die Verknüpfung von den Daten

der darunterliegenden Schicht mit Metadaten geschieht. Die eigentlichen Daten werden also mit Informationen über sie selbst „angereichert“. Diese Metadaten werden z.B. in Datenbanken gespeichert. Darüber hinaus kann die Nutzung eines Persistent Identifiers sinnvoll sein, also einer ID, die – einmal vergeben – ein eindeutiges permanentes Charakteristikum eines Datums wird. So wird es auch möglich, den physikalischen Speicherort von Daten in Schicht 1 zu verlegen, die Objektverwaltung aber zu zentralisieren. Die oberste, für Nutzerinnen und Nutzer sichtbare Schicht 3 ist die Nutzungsschicht. Hier können Daten eingepflegt, aber auch Nutzerrollen angepasst und nach Daten gesucht werden. Die Tools auf dieser Ebene müssen in den wissenschaftlichen Wertschöpfungszyklus eingebettet werden können. Diese Aufteilung entspricht im Wesentlichen dem Drei-Schichtenmodell im allgemeinen Software-Entwurf (Ludewig und Lichter, 2013, 430f). Die unterste Schicht wird als Datenhaltungsschicht bezeichnet und dient der persistenten Datenhaltung, die zweite Schicht ist die Anwendungsschicht und stellt die fachliche Funktionalität eines System dar. Die oberste Schicht ist die Präsentationsschicht, welche die Bedienoberfläche bereitstellt.

Im Ansatz von (Askhoj, Sugimoto und Nagamori, 2011) wird das rein funktional gedachte OAIS-Modell auf eine Schichtenarchitektur abgebildet, um diese kompatibel mit dem Cloud-Gedanken zu machen, d.h. einzelne Funktionen bzw. Schichten auf entfernte oder externe Dienstleister auszulagern. Problematisch wird die Interdependenz zwischen funktionalen Elementen in OAIS gesehen, was dazu führt, dass einige Funktionen über mehrere Schichten abgebildet werden müssen. Die Autorinnen und Autoren gehen von vier nötigen Schichten aus. Dies ist zunächst eine PaaS (Platform as a Servicer)-Schicht, die die persistente Speicherung der Objekte übernimmt. Die darauf aufbauende SaaS (Software as a Service)-Schicht ist für die Verwaltung der Daten zuständig und macht sie zu Objekten, indem hier deskriptive Metadaten sowie ein Link auf den Speicherort (in der PaaS-Schicht) abgelegt werden. Die Preservation-Schicht baut aus den Daten der PaaS-Schicht und den deskriptiven Metadaten der SaaS-Schicht mit weiteren Metadaten, wie Formatrepräsentationsinformationen oder technischen Metadaten (den Preservationsinformationen), Datenpakete, die von der Interaction-Schicht dann herausgegeben werden können. Letztere fungiert als Zugriffsschicht. Die Abbildung von OAIS auf ein Schichtenmodell sowie die praktische Trennung von Metadaten und Daten in verschiedenen Schichten ist von hoher Bedeutung für die vorliegende Dissertation. Allerdings ist das System nur für die Verwaltung von E-Mails getestet worden. Auch ist fraglich, ob Cloud-Dienste zur Speicherung von sehr großen Datenmengen überhaupt genutzt werden können.

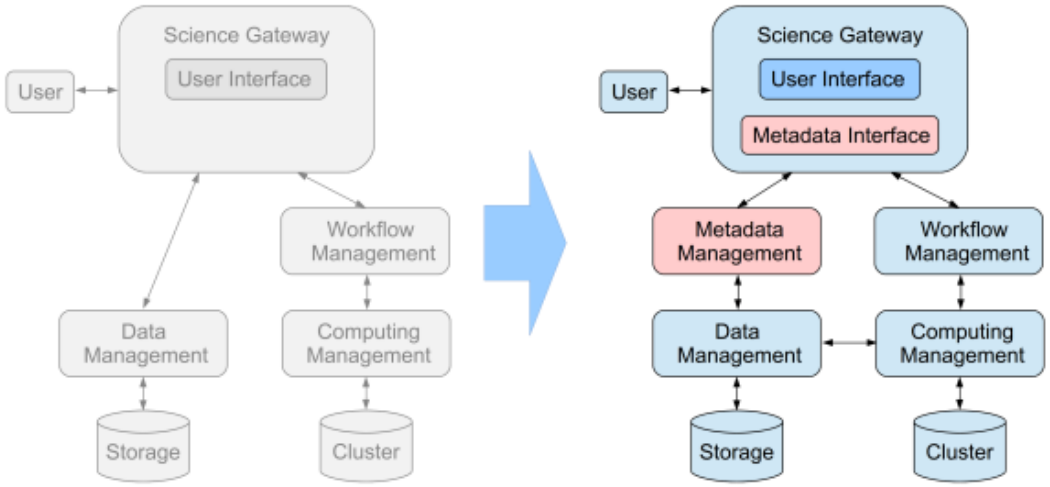

Abbildung 2.5: Der abstrakte Arbeitsprozess, der um eine generische Metadatenmanagementkomponente ergänzt ist, entspr. (Grunzke, 2016, S. 63).

2.4 Ansätze in angrenzenden Bereichen

2.4.1 Ansätze in der Molekulardynamik

In seiner Dissertation „Generic Metadata Handling in Scientific Data Life Cycles“ (Grunzke, 2016) stellt Richard Grunzke einen generischen Ansatz zum Forschungsdatenmanagement vor, den er für die Molekulardynamik konkretisiert. Der simulationswissenschaftliche Arbeitsprozess und somit auch der Datenlebenszyklus kranke an einem Fehlen von Metadatenmanagement. Diese Analyse liefert die Begründung für die Einführung einer weiteren abstrakten, technologieunabhängigen Komponente, nämlich der des Metadatenmanagements (und eines entsprechenden Interfaces), was in Abbildung 2.5 dargestellt ist. Diese Kompontente soll möglichst automatisierbar sein, um sich nahtlos in den Arbeitsprozess einfügen zu lassen. In der Dissertation (Grunzke, 2016) wird diese Abstraktion als wesentlich herausgestellt, um möglichst generisch bleiben zu können und die Komplexität der Datenhaltung zu beherrschen, wodurch der Ansatz zunächst unspezifisch bleibt. Für die konkrete Ausgestaltung eines Arbeitsprozesses und die Definition eines Datenlebenszyklus wird in Kapitel 3.4.1 in (Grunzke, 2016, 68f) ein Planungsvorschlag geliefert, welcher auch in (Gesing u. a., 2015) veröffentlicht wurde und Fragen zu den Anforderungen umfasst. Außerdem werden einige allgemeine Technologieempfehlungen zum Daten- und Metadatenmanagement ausgesprochen, die insbesondere die Nutzung von UNICORE[47] empfehlen.

[47] `https://www.unicore.eu/`, Zugriff 3.1.2019. Unicore stellt eine Grid-Middleware zur Verbindung von Höchstleistungsrechenzentren auf europäischer Ebene dar und dient als Abstraktionsschicht von Supercomputerspezifika. Mit dem Storage Management Service wurde eine Abstraktionsschicht eingeführt, die von der jeweiligen konkreten Speichertechnologie abstrahieren kann und zum Distributed Storage Management Service weiterentwickelt wurde (Rekawek, Bała und Benedyczak, 2011).

Schließlich wird eine solche konkretisierte Variante für den Anwendungsfall der Molekulardynamik implementiert, welche einen konkreten Datenlebenszyklus in der computergestützten Chemie darstellt. Eingebettet ist das Vorhaben in das Molecular Simulation Grid (MoSGrid)[48], eine Plattform zur systemunabhängigen Definition von Arbeitsprozessen in der Molekulardynamik. Dabei wird die Molecular Markup Simulation Language (MSML), eine auf MoSGrid angepasste Variante der Chemical Markup Language (CML), verwendet (Murray-Rust und Rzepa, 2011; Grunzke u. a., 2014). Diese hierarchische XML-Datenbeschreibung wird nun von Wissenschaftlerinnen und Wissenschaftlern zur Definition eines Workflows benutzt, woraus dann die, für einzelnen Simulationscodes spezifischen, Eingabedateien erstellt werden. Die Wissenschaftlerinnen und Wissenschaftler erstellen also nicht mehr die spezifischen Eingabedateien für die spezifischen Simulationscodes, sondern, eine logische Ebene höher, eine abstraktere Beschreibung, die jedoch wieder aus einem gewissen programmspezifischen Vokabular besteht. Nach dem Simulationslauf finden sich im MSML-Dokument dann Informationen wie die Art der Moleküle, die simuliert wurden, die Lösungsmittel und Kraftfelder, die Elektrostatik und die van der Waals-Kräfte, Temperatur und Druck, die benutzten Anwendungen für die Vorbereitung und Durchführung der Simulation, die genutzten Rechenressourcen, die generierten Outputstrukturen und deren Charakteristiken. Dabei nimmt das MSML-Dokument die zentrale Stelle ein, indem es einerseits als Informationsprovider für die Simulation selbst fungiert, andererseits auch die nachträgliche Metadatenextraktion und -annotation ermöglicht. Die Metadatenextraktion wurde innerhalb der Arbeit von (Grunzke u. a., 2014) für die Datenformate PDB und SDF implementiert. Die Metadatenextraktion überführt schließlich Information nach dem vollendeten Simulationslauf aus dem MSML-Dokument, welches geparste und eingetragene Informationen enthält, in das JSON-Datenformat und registriert die Daten und Metadaten in UNICORE. Die Metadatenextraktion ist mit JAVA SAX implementiert.

Die Gemeinsamkeit zwischen dem Ansatz von (Grunzke u. a., 2014) und der vorliegenden Arbeit liegt in dem Fokus auf das Metadatenhandling und dessen Integration in den Arbeitsprozess. Dass Metadatenmanagement als kritische Komponente in den wissenschaftlichen Arbeitsprozess eingeführt werden muss, steht außer Frage und findet sich auch im vorliegenden Dissertationsvorhaben wieder. Dies zeigt sich sowohl in den Anforderungen der Metadaten (Kapitel 4.1.1) und der Integration in den Arbeitsprozess (Kapitel 4.1.2), als auch in der Umsetzung von *EngMeta* als Metadatenmodell und der automatisierten Erfassung von Metadaten.

Die Ansätze unterscheiden sich allerdings gravierend in der Herangehensweise und Durchführung. Da die vorliegende Arbeit neben der fachspezifischen und computerwissenschaftlichen auch die informationswissenschaftliche Seite integriert, wurde *EngMeta* aus bestehenden allgemeinen Metadatenmodellen entwickelt, wohingegen der Ansatz von MSML auf CML basiert. Die Anlehnung an allgemeine, nicht disziplinspezifische Metadatenmodelle hat den Vorteil, dass diese Teile später auch

[48] `https://www.mosgrid.de/portal`, Zugriff 5.4.2019.

wieder leicht zu extrahieren sind, was die Kompatibilität mit Standards garantiert. Dies ist insbesondere wichtig, um die Daten problemlos in Repositorien einspielen zu können oder um einen persistenten Identifier zu erhalten (bspw. basiert DOI auf DataCite). Des Weiteren ist MSML auf die Molekulardynamik beschränkt, wohingegen EngMeta allgemein für viele computergestützte Ingenieurwissenschaften konzipiert wurde und so offener gestaltet ist. Der wesentliche Unterschied zwischen MSML und *EngMeta* ist jedoch die Art der Einbettung in den Arbeitsprozess. Dabei ist MSML eng mit der MosGrid-Plattform zur Definition von Arbeitsabläufen verbunden. Dies bedeutet, dass die Vorteile der Metadatendokumentation nur genutzt werden können, wenn mit der Plattform gearbeitet wird. Vom Standpunkt der organisatorischen Herausforderungen, die ich in Kapitel 1.3.2 herausgearbeitet habe, ist es fraglich, ob die Wissenschaftlerinnen und Wissenschaftler den Zusatzaufwand der Definition des Workflows im MSML-Dokument mittels der MosGrid-Plattform in Kauf nehmen, um von den Vorteilen zu profitieren[49]. Das, in der vorliegenden Dissertation beschriebene, Vorgehen setzt logisch gesehen weiter unten im Arbeitsprozess an und unterstützt die Wissenschaftlerinnen und Wissenschaftler darin, nach Beendigung des Simulationsprozesses die Metadatenannotation möglichst automatisiert durchzuführen.

Im Bereich der Molekulardynamik wird auch im Rahmen des European Materials Modelling Council (EMMC) an *Model Data* (MODA) gearbeitet[50]. MODA soll die Modellierung von Arbeitprozessen standardisieren und zentral aufnehmen. An einer Formalisierung als Ontologie wird derzeit gearbeitet.

Als weitere Entwicklung ist die MolMod-Datenbank[51] (Stephan u. a., 2019) zu nennen, die als Open Access-Datenbank Kraftfelder von 150 Molekülen enthält. Diese Kraftfelder werden dort in einer standardisierten Weise mit ihrer Herkunft angegeben und können von dort automatisch in Eingabedateien für typische Simulationscodes der Molekulardynamik exportiert werden, u.a. in das GROMACS-Eingabeformat. Da Fehler in den Eingabedateien als eine der größten Fehlerquellen genannt werden, soll die Standardisierung durch die MolMod-Datenbank Abhilfe schaffen.

2.4.2 Ansätze in den Klimawissenschaften

Forschungsdatenmanagement in den Klimawissenschaften hat besondere Relevanz, da Beobachtungs- und Modelldaten aus Satellitenaufnahmen bzw. Simulationsläufen gesellschaftliche Implikationen haben können. Beispielsweise können die Rechnungen, die für das IPCC (Intergovernmental Panel on Climate Change) ausgeführt werden, als Grundlage für politische Entscheidungen dienen. Dabei werden Zirkulationsmodelle verwendet, die global mehrere Modelle, z.b. für Atmosphäre oder Ozeanographie, aneinander koppeln (Ludwig und Enke, 2013). Die Simulationen

[49] Eine weitere Idealisierung in (Grunzke u. a., 2014), auf die hier nur am Rande eingegangen werden soll, ist die, dass sich Workflows und damit auch Simulationen überhaupt plattformunabhängig definieren lassen.

[50] `https://emmc.info/moda/`, Zugriff 4.4.2019

[51] `http://molmod.boltzmann-zuse.de`, Zugriff 3.5.2019

innerhalb des *Coupled Model Intercomparison Project* (CMIP) (mittelerweile in Stufe 6) finden unter internationaler Aufsicht des World Climate Research Programme (WCRP)[52] statt (Eyring u. a., 2016). Dort sind hohe Standards an die Datenhaltung gesetzt, z.B. eine festgelegte Benennung der Daten, Datendokumentation, NetCDF als interkompatibles Datenformat mit Metadaten nach CF-Konvention, persistente Identifier (DOI) und konsistente Zitierrichtlinien für Daten (Stockhause und Lautenschlager, 2017). Diese Daten werden öffentlich zur Benutzung bereit gestellt. Die Bereitstellung findet u.A. durch das Deutsche Klimarechenzentrum (DKRZ) statt[53], von dem auch das World Data Center for Climate (WDCC)[54] betrieben wird. Das WDCC ist ein Teil des Erdsystem-Datenverbunds ESGF[55] und stellt einen weltweiten Dienst zur Langzeitarchivierung und Veröffentlichung von klimawissenschaftlichen Daten, darunter auch die Daten der IPCC Simulationsläufe, bereit. Bis 2020 sollen im WDCC 500 PB Daten gehalten werden. Mit CERA2.5 steht ein Metadatenmodell[56] zur Verfügung, um diese Daten auch fachwissenschaftlich standardisiert beschreiben zu können.

Es zeigt sich, dass die Klimawissenschaften bereits für einige Fälle (im Rahmen der IPCC-Modellrechnungen) strenge Standards – sowohl technische als auch organisatorische – etabliert haben. Dieses Standards haben extrinsische und intrinsische Gründe: Einerseits müssen Vorgaben von Kommissionen oder Gesetzgebern betrachtet werden, andererseits dürfen sich die Klimawissenschaften nicht angreifbar machen, und müssen methodisch transparent arbeiten. Diese Standards wirken auch in die Arbeit von Klimawissenschaftlerinnen und -wissenschaftlern hinein, die nicht innerhalb der o.g. Projekte arbeiten. Ebenso können die Klimawissenschaften durch die genannten Ansätze potentiell eine Vorbildfunktion für andere Wissenschaftszweige einnehmen.

2.4.3 Ansätze in der Hochenergiephysik

Neben den Klimawissenschaften ist die Hochenergiephysik die Disziplin mit der größten Nutzung von Höchstleistungsrechnerressourcen und mit der größten Datenproduktion. Schon früh wurden Anforderungen formuliert, die sich schließlich im Gedanken des Grid-Computing kristallisierten (Foster und Kesselman, 1999). Für die Hochenergiephysik müssen sehr große Datenmengen, verteilt über ein lose gekoppeltes Rechnernetz, gespeichert und analysiert werden können. Diesem Umfeld entstammt auch GridFTP als performantes, paralleles Übertragungsprotokoll zum schnellen Austausch von großen Datenmengen in Rechnernetzen (Allcock u. a., 2002).

Die ATLAS-Experimente, in deren Rahmen auch die Existenz des Higgs-Boson bestätigt wurde, verwenden Xcache (Hanushevsky u. a., 2018; Yang u. a., 2018) und RUCIO[57] (Barisits u. a., 2017) zum Datenmanagement. Xcache basiert auf dem

[52] `https://www.wcrp-climate.org/wgcm-cmip/wgcm-cmip6`, Zugriff 24.4.2019.
[53] `https://cmip-esmvaltool.dkrz.de/`, Zugriff 24.4.2019.
[54] `https://cera-www.dkrz.de/WDCC/ui/cerasearch/`, Zugriff 24.4.2019.
[55] ESGF = Earth System Grid Federation, siehe `https://esgf.llnl.gov/`, Zugriff 24.4.2019.
[56] Siehe Kapitel 2.2.2.
[57] `https://rucio.cern.ch/`, Zugriff 3.5.2019.

Xrootd-Protokoll[58] und dient dem asynchronen Abrufen und Cachen von Daten im verteilten Datenmanagementnetzwerk RUCIO. Mittels RUCIO werden die Daten zusammengehöriger Experimente, die auf verschiedenen Teilchenbeschleunigern an unterschiedlichen Orten durchgeführt werden, verwaltet, wobei die tatsächlichen physikalischen Speicherorte weltweit verteilt sind. Im Rahmen von ATLAS werden 400 PB Daten in 1 Milliarde Dateien auf 130 Einrichtungen weltweit verteilt gehalten (Barisits, 2018). RUCIO verwaltet Dateien, die sich zu Datensätzen, und diese wiederum in Containern, organisieren lassen und so beispielsweise gemeinsam mittels Xcache an einen spezifischen Ort geholt werden können. Allerdings verwaltet es die Daten, Datensätze und Container in einem gemeinsamen Namensraum, der ortstransparent ist. Die Daten können hierbei regelbasiert verschoben werden. Bzgl. Metadaten bietet RUCIO die Möglichkeit, Datensätze mit Metadaten zu annotieren, die gemäß eines vorgegebenen Vokabulars als *Schlüssel-Wert*-Paare hinterlegt werden, wobei JSON genutzt werden kann[59]. Der Fokus bei RUCIO liegt, den Anforderungen der Hochenergiephysik entsprechend, allerdings mehr im verteilten Management von großen Daten und nicht auf einem detaillierten Metadatenmanagement.

2.5 Relevante Datenmanagementsysteme im HPC-Umfeld

2.5.1 iCurate

In (Liang u. a., 2015) wird iCurate vorgestellt, ein Datenmanagementsystem an der Universität von Huddersfield, UK, welches auf HPC-Daten spezialisiert ist und semantische Annotation unterstützt. Hierfür können die Nutzerinnen und Nutzer ihre Metadaten in der PBS-Job-Datei deklarieren. Dies geschieht so, dass jegliche Kommandos, z.B. der Start der Simulation oder die Angabe des Skripts, in Direktiven eingeschlossen werden, die später vom Auswertungsprozess aufgenommen werden können. Bei iCurate werden dann Snapshots eines ganzen Verzeichnisses erstellt, die mit den entsprechenden Metadaten versehen werden. Dem System liegen die Metadatenstandards PREMIS und OAIS zugrunde.

Zwar ist die Aufnahme von Metadaten unmittelbar nach bzw. bei der Erzeugung der Daten ein mächtiges Werkzeug, allerdings werden nur Prozessmetadaten automatisiert erfasst (z.B. Start-, End- und Laufzeit der Simulation). Ein generisches Parsing, welches auch die Extraktion von fachspezifischen Metadaten aus den Logfiles der Simulationscodes umfasst, ist mit diesem Ansatz nicht möglich.

[58] `http://www.xrootd.org`, Zugriff 3.5.2019.

[59] Für die ATLAS-Experimente selbst wurde mit COMA (Conditions/ConfigurationMetadata for ATLAS) (Gallas u. a., 2012) eine fachspezifische Datenbank zur Beschreibung der Experimente entwickelt.

2.5.2 iRods

iRods[60] (Integrated Rule-Oriented Data System) geht auf die Ansätze des Grid-Computing zurück und basiert konzeptionell auf dem Vorgängersystem Storage Resource Broker und bietet ein regelbasiertes, verteiltes Datenmanagement-System. Dabei bildet es einen eindeutigen Namensraum für Datenobjekte, die auf verschiedene Datenträgerarten und/oder auf verschiedene Orte verteilt gespeichert sind. Mittels Regeln lassen sich Operationen definieren, die Daten z.B. automatisiert von einem Ort an den anderen verschieben, wenn gewissen Bedingungen eintreten. iRods verfügt auch über Metadatenmanagement, was in neueren Implementierungen auf einen eigenen Server ausgelagert wurde, wobei Metadaten intern als Tripel (Key, Value, Unit) gespeichert werden. iRods wird von vielen Hoch- und Höchstleistungsrechenzentren zum Datenmanagement verwendet, so z.B in Finnland vom CSC[61] oder in Frankreich vom IN2P3[62]. Auch wurde es beispielsweise erfolgreich genutzt, um 500 TB Genomsequenzdaten für die Krebsforschung über verschiedene Datenablageorte zu verwalten (Chiang u. a., 2011).

2.5.3 signac

Signac (Adorf u. a., 2018) entstammt dem HPC-Umfeld und bietet lokales Datenmanagement für Dateisysteme und Workspaces. Es ist im Kern eine Datenbank, die den Namensraum des Dateisystems abstrahiert, Metadaten mitführt und auch über Workspace-Grenzen hinweg funktioniert. Die Datenobjekte selbst, egal ob ein Datensatz oder einzelne Werte, werden auf dem Dateisystem verwaltet, wohingegen die weiteren Informationen, wie Metadaten oder eine universelle Namensbezeichnung, in der Datenbank zur Suche mitgeführt werden. Die Metadaten werden hierbei in JSON abgelegt. Dadurch muss bei der Suche nach Datenobjekten nicht das Dateisystem durchsucht, sondern nur eine Anfrage an die Datenbank gestellt werden. Zur Nutzung wird ein Python-Paket angeboten (welches z.B. im Molekulardynamik-Simulationscode HOOMD-blue genutzt werden kann), aber auch eine Schnittstelle, um Tools miteinzubeziehen, die nicht in Python geschrieben sind. Das Kommandozeilen-Tool wird hierbei direkt aus dem jeweiligen Code heraus aufgerufen. Somit stellt signac eine niederschwellige Lösung dar, die direkt am Prozess ansetzt und lokales Datenmanagement zum Ziel hat. Abstrakte Datenobjekte können in einer Datenbank verwaltet werden, ohne dabei auf einen spezifischen Simulationscode oder eine spezielles Metadatenmodell zugeschnitten zu sein.

[60] `https://irods.org/`, Zugriff 28.2.2019.
[61] `https://research.csc.fi/csc-guide-archiving-data-to-the-archive-servers`, Zugriff 28.2.2019.
[62] `https://irods.in2p3.fr/`, Zugriff 28.2019.

2.6 Diskussion der Vorarbeiten

Die Vorarbeiten zu Metadaten werden in der Promotion genutzt, wobei hier speziell auf den Definitionen nach (Edwards u. a., 2011) von Metadaten als Kommunikationsform und nach (Greenberg, 2003) von Metadaten als strukturierte Beschreibung und Unterstützung des Arbeitsprozesses aufgebaut wird. Die besprochenen Metadatenmodelle bilden später die technische Grundlage für die Entwicklung von *EngMeta*, die in Kapitel 5 vollzogen wird. Die FAIR-Prinzipien gehen als genereller Orientierungsrahmen für gutes Forschungsdatenmanagement in die Dissertation ein, wobei sich damit nicht die Fragen nach der Datenverantwortlichkeit lösen lassen. Die FAIR-Prinzipien stehen vor allem in ihrer Erweiterung (EU, 2018) für Forschungsdatenmanagement als Zusammenspiel von Technologien (Metadatenannotation, persistente Identifier, Repositorien, Dienste), organisatorischen Maßnahmen (Forschungskultur, Ausbildung, Anreize) und Förderung (finanzieller und ideeller Art). Das OAIS-Modell und die weiteren vorgestellten Schichtenmodelle dienen als Blaupause für ein Schichtenmodell für Forschungsdatenmanagement, welches sich in Kapitel 5.1 umgesetzt findet. Die Ansätze in der Molekulardynamik schienen zunächst ähnlich zum Vorhaben der Dissertation. Sie enthalten beide eine Plattform für Forschungsdatenmanagement, ein Metadatenmodell zur Datenbeschreibung sowie eine automatisierte Metadatenerfassung. Allerdings ist der Ansatzpunkt im Arbeitsprozess ein anderer. In der vorgestellten Arbeit müssen die Wissenschaftlerinnen und Wissenschaftler ihren Arbeitsprozess über das Forschungsdatenmanagementsystem definieren. Die automatisierte Metadatenerfassung extrahiert die, in dieser Definition des Arbeitsprozesses hinterlegten, Metadaten. Da die Hemmschwelle der Nutzung als zu hoch eingeschätzt wurde, wird dieser Ansatz in der Dissertation nicht weiter aufgenommen. Außerdem ist die Anwendung auf die Molekulardynamik beschränkt, wohingegen die Dissertation den Anspruch verfolgt, Metadatenbeschreibung und -annotation für sämtliche computergestützten Ingenieurwissenschaften zu leisten. Bzgl. der Datenmanagementsysteme wurden Elemente von bestehenden Systemen in das Lösungskonzept integriert. So wurde beispielsweise von iCurate die Idee erweitert, dass semantische Annotation bereits in den Job-Dateien der Simulation stattfindet und diese später ausgewertet wird. iCurate prüft die bzgl. enthaltener Informationen wertvollen Log-Dateien der Simulationscodes allerdings nicht. Signac könnte aussichtsreich sein für eine spätere Entwicklung einer lokalen Datenbank für die jeweiligen Nutzerinnen und Nutzer. Diese Idee konnte aber nicht weiterverfolgt werden, da die Priorität innerhalb der Dissertation auf der Prozessintegration durch eine automatisierte Metadatenerfassung mit Anbindung an ein Repositorium lag, um den Pfad zu Open Access wenigstens prinzipiell aufzuzeigen. iRods wurde nicht als Grundlage für das FDM-System genutzt, da die Repositoriumsfunktionalitäten als zu wenig ausgeprägt ermittelt wurden und das RUCIO-System wurde als zu spezifisch für die Hochenergiephysik eingeschätzt.

Kapitel 3

Dunkle Daten im Höchstleistungsrechnen

Die, in Kapitel 1.3 diskutierten, Herausforderungen gipfeln in einem Phänomen, welches ich – in Anlehnung an die bestehenden Vorarbeiten, auf die in Kapitel 2.1 bereits kurz eingegangen wurde – im Folgenden *dunkle Daten* nenne. Dunkle Daten sind Daten, deren Sichtbarkeit und damit Nutzbarkeit eingeschränkt ist, indem sie nicht zugreifbar und/oder nicht ausreichend beschrieben sind, obwohl sie einmal, mit unter Umständen erheblichem technischem, wissenschaftlichem und finanziellem Aufwand, erstellt wurden. Dunkle Daten werden bisher insbesondere von Unternehmen im Zusammenhang mit Sensordaten thematisiert: Bei Sensordaten gelten bis zu 90 % der erfassten Daten als dunkel[63], d.h. als nicht nutzbar oder nicht genutzt. Eine neuere Studie von IBM kommt zu dem Ergebnis, dass bis zu 80 % der Daten von Unternehmen dunkel sind[64], da sie unstrukturiert vorliegen und damit ihre Verarbeitungsmöglichkeiten stark eingeschränkt sind. Im Bereich der Wirtschaft und der Unternehmen werden dunkle Daten zusehends zum Thema gemacht, insbesondere vor dem Hintergrund, dass Unternehmen nun in der Lage sind, *alle* Daten, gleich ihrer Struktur, Qualität, Wichtigkeit usw. zunächst ohne konkretes Ziel zu sammeln und später auszuwerten. Dies markiere einen Paradigmenwechsel von „no data“ zu „dark data“ (Cafarella u. a., 2016).

Eine generelle „Verfallskurve“ von Wissen über Forschungsdaten (Michener u. a., 1997, S. 332) wird in Abbildung 3.1 gezeigt. Diese Verfallskurve spiegelt die Folgen der, in Kapitel 1.3 analysierten Herausforderungen, wider. Zum Zeitpunkt der Publikation ist das Wissen über die Daten am größten und nimmt über die Zeit stetig ab, wobei immer mehr Details verloren gehen, sofern keine Dokumentation vorhanden ist. Das Wissen über die Daten geht durch Ereignisse wie den Verlust der Daten selbst, deren Dokumentation, sowie personelle Wechsel noch schneller verloren und lässt Daten dunkel werden.

---

[63] `https://siliconangle.com/blog/2015/10/30/ibm-is-at-the-forefront-of-insight-economy-ibminsight/`, Zugriff 26.6.2018.

[64] `https://www.ibm.com/blogs/think/be-en/2018/04/24/marketing-dark-dark-data/`, Zugriff 2.3.2019.

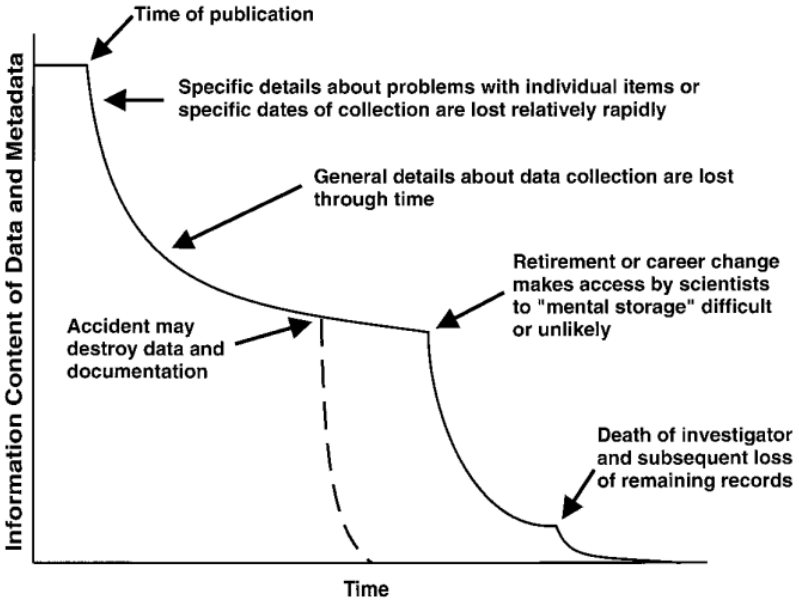

Abbildung 3.1: Verfallskurve von Wissen über Forschungsdaten (Michener u. a., 1997, S. 332).

In der wissenschaftlichen Literatur werden dunkle Daten von zwei Autoren thematisiert[65]. Auf die Ausführungen von Thomas Goetz (Goetz, 2007) kann nicht weiter eingegangen werden, da sich meine Analyse von dunklen Daten weniger auf den Inhalt (hier: ein erfolgreicher oder fehlgeschlagener Simulationslauf) der Daten bezieht als auf die Form. Negative Forschungsergebnisse sind nicht das Charakteristikum von dunklen Daten und werden daher unter dem Aspekt von Datenunverfügbarkeit generell betrachtet, gleich ob es sich inhaltlich um erfolgreiche oder fehlgeschlagene Simulationsläufe handelt. Bryan Heidorns (Heidorn, 2008; Heidorn, Stahlman und Steffen, 2018) Definition erfasst lediglich die Form der Daten (verfügbar/unverfügbar) und liegt den weiteren Überlegungen zugrunde und wird im nächsten Unterkapitel auf das Höchstleistungsrechnen erweitert.

3.1 Definition Dunkler Daten für das Höchstleistungsrechnen

Bryan Heidorns Definition zufolge handelt es sich bei Daten um dunkle Daten, wenn sie nicht sorgfältig indiziert[66], d.h. nicht ausreichend mit Metadaten versehen und/oder nicht sorgfältig gespeichert sind:

[65] In (Purss u. a., 2015) werden dunkle Daten im Bereich der Satellitendaten rein phänomenologisch thematisiert. Rohdaten, die als dunkle Daten bezeichnet werden, wurden hier von Bandarchivsystem in eine kalibrierte Form überführt.

[66] „not carefully indexed“ bzw. „nicht sauber indiziert“ kann von der Begrifflichkeit her falsch verstanden werden, da sich der Begriff „Index“ oft auf die Indexstruktur von Datenbanken bezieht. Der Begriff Datenindizierung wird hier aber im Sinne von Metadatenannotation verwendet.

> „'Dark data' is not carefully indexed and stored so it becomes nearly invisible to scientists and other potential users and therefore is more likely to remain underutilized and eventually lost." (Heidorn, 2008, S. 280)

Somit treten dunkle Daten aus zweierlei Gründen auf. Einerseits, weil Daten „nicht sauber indiziert", d.h. nicht mit ausreichend Metadaten versehen sind, um sie nachnutzen zu können. Außerdem treten laut Heidorn dunkle Daten auf, indem sie „unsichtbar" werden, weil die physikalische Speicherung der Daten nicht nachhaltig gestaltet ist. Als Beispiel führt er hier portable Medien wie USB-Sticks und Wechselfestplatten an, die im Büro verschwinden können:

> „[...] the type of data that exists only in the bottom left-hand desk drawer of scientists on some media that is quickly aging and soon will be unreadable by commonly available devices" (Heidorn, 2008, S. 281)

Dabei können sich beide Formen überschneiden und schließen sich nicht aus, bedingen sich aber auch nicht gegenseitig. Mit Metadaten versehene Daten können trotzdem verschwinden, weil – um im originalen Beispiel zu bleiben – z.B. der USB-Stick unauffindbar ist. Vice versa können Daten in definierten Festspeichern auf Dateisystemen verfügbar, aber ohne Metadaten versehen ungenutzt bleiben, weil sie nicht verstanden werden können. Ebenso können sie ohne Metadaten versehen sein und auf einer Wechselfestplatte in einer Schreibtischschublade verloren gehen.

Seine Begriffsdefinition stützt Heidorn auf die Analyse von Daten aus kleinen Forschungsunternehmungen, dem *long tail of science*[67], denen keine oder nur wenig Zeit, Interesse und finanzielle Mittel zum Datenmanagement nachgewiesen werden. In einem Folgeartikel (Heidorn, Stahlman und Steffen, 2018) weist Heidorn dunkle Daten konkret im *long tail* der Astronomie nach. Er sieht in vielen kleineren Forschungsprojekten eine erhöhte Gefahr für das Auftreten von dunklen Daten, welche er typischerweise durch folgende Probleme befördert sieht:

> „Dark data in the long tail are typically: heterogeneous; hand generated; created through unique procedures; curated by individuals; often archived in personal or institutional repositories; not maintained; obscured or protected; seldom reused; and currently unnoticed." (Heidorn, Stahlman und Steffen, 2018, S. 4)

Im Folgenden versucht er, seine Trennung in *head* und *long tail* bzw. Big Data und Small Science in der Astronomie aufrechtzuerhalten, indem er schreibt:

[67] Der Begriff *long tail of science* wurde in Anlehnung an den Begriff *long tail* (Anderson, 2004) der Internetökonomie gebildet, durch den ausgedrückt wird, dass diese weniger von Massenprodukten als von vielen kleinen Nischenprodukten getrieben wird. In Heidorns Sinne bezeichnet der Begriff den Großteil der (kleinen) wissenschaftlichen Projekte, die die wissenschaftliche Erkenntnis vorantreiben. Diese seien dadurch charakterisiert, dass deren finanzielle Mittel im Vergleich zu wissenschaftlicher Großforschung (welche er als *head* bezeichnet) gering sind. Der *long tail of science* bestimmt sich also über eine Verteilungskurve, die das Verhältnis von der Höhe der Forschungsmittel zu ihrer Anzahl der Größe nach sortiert anzeigt.

> „While astronomy is increasingly a "big data" field, "little" or "small" science which is the primary focus of the Long Tail theory is typically hypothesis-driven [...]. [...] rapid generation of data in astronomy do not necessarily preclude the ongoing presence of small science in astronomy, where scientists may generate and hold innovative data in private collections." (Heidorn, Stahlman und Steffen, 2018, S. 4)

Indem Heidorn feststellt, dass die Astronomie eigentlich Big Data produziert, aber die Forschung in kleinere Einheiten zerteilt stattfindet, versucht er die analytische Trennung festzuschreiben.

Wie sich HPC überhaupt im Sinne von Heidorn einordnen ließe, ist für die Begrifflichkeit problematisch. Wenn – Heidorn folgend – nur die finanziellen Mittel betrachtet würden, wären die einzelnen Rechenprojekte tatsächlich eher dem *long tail of science* zuzuordnen. Werden die Kosten der Höchstleistungsrechner sowie des Personals mit einbezogen, wäre die computergestützte Forschung mit HPC eher dem *head* zuzuordnen. Im Sinne der Datenproduktion ist die Bestimmung aber ganz eindeutig im *head* bzw. in der Head Data vorzunehmen. Die Bestimmung, was *head* und *long tail* bzw. *head data* und *tail data* ist und eine Grenze zu ziehen, wie es Heidorn über die Forschungsmittel vornimmt, ist fragwürdig. Die Trennung in *head* und *tail* ist also schwierig im HPC aufrechtzuerhalten, ebenso die Angabe der Forschungsmittel. Selbst, wenn davon ausgegangen wird, dass es sich bei computergestützter Forschung mittels HPC um den *head* bzw. *head data* handelt, hält die Idealisierung dieses Bereichs, die Heidorn vornimmt, nicht stand. Datenmanagement im HPC ist nicht besser organisiert, wie in der in den vorhergehenden Kapiteln 1.2 und 1.3 vorgelegten Analyse gezeigt wurde. Vor allem in den computergestützten Ingenieurwissenschaften sind die Daten – zumindest außerhalb der Rechenprojektgrenzen – nicht homogen sowie durch spezifische Softwarecodes generiert. Standardisierte Metadatenmodelle fehlen, zentrale Repositorien sind meist nicht vorhanden und das Datenmanagement beschränkt sich auf Speicherung auf beliebigen Dateisystemen und Pflege durch einzelne Individuen. Anreize zum Forschungsdatenmanagement fehlen für Wissenschaftlerinnen und Wissenschaftler und der Sinn ist oft nicht klar ersichtlich. Dies entspricht im Wesentlichen den Charakteristiken für Daten aus einem Bereich, die im Zitat oben festgestellt wurden, nämlich „*heterogeneous; [..] created through unique procedures; curated by individuals; often archived in personal or institutional repositories; not maintained; obscured or protected; seldom reused*" (Heidorn, Stahlman und Steffen, 2018, S. 4).

Die Verknüpfung des Konzepts der dunklen Daten mit der Art der Wissenschaft bzw. der Größe der Wissenschaftsvorhaben (*head* vs. *long tail*), die von Heidorn aufgestellt wurde, wird im Folgenden für HPC verworfen, da für die computergestützte Forschung mittels Höchstleistungsrechnern – wie im obigen Absatz diskutiert – ebensolche Probleme nachgewiesen werden konnten und Heidorns Idealisierung vom Datenmanagement im *head* eine Fehldiagnose darstellt.

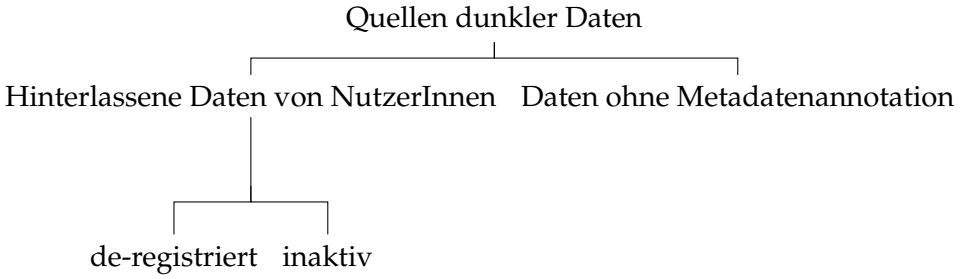

Abbildung 3.2: Taxonomie der dunklen Daten im HPC. Einerseits entstehen dunkle Daten durch fehlende Metadaten, andererseits durch de-registrierte oder inaktive Nutzerinnen und Nutzer. Die Taxonomie wurde vorab publiziert in (Schembera und Durán, 2019).

Da die Definition von dunklen Daten sich allein aus der Höhe von Forschungsmitteln als nicht haltbar erwiesen hat, wird eine Definition dunkler Daten für HPC vorgeschlagen, die sich allein als Formbestimmung darstellt:

> Dunkle Daten sind Daten auf persistenten Festspeichern in Rechenzentren, die durch fehlende Metadatenannotation oder Inaktivität der Nutzerinnen und Nutzer entstehen. Sie sind dadurch nicht mehr verstehbar oder zugreifbar und in letzter Instanz nicht mehr nutzbar. Dabei können sie unabhängig von der Speichertechnologie (SSD, Festplatten, Bandspeicher) entlang der kompletten Speicherhierarchie (Home-Dateisystem, paralleles Dateisystem, HSM-System, Backup-System, Archiv-System, Datenbanken) entstehen.

Es wird nun gezeigt, wodurch dunkle Daten im HPC auftreten können. Im HPC speisen sich dunkle Daten aus zwei Hauptquellen, die in Abbildung 3.2 dargestellt sind (Schembera und Durán, 2019). Einerseits entstehen dunkle Daten im HPC durch fehlende Metadatenannotation bzw. Unterindizierung. Andererseits entstehen sie durch Hinterlassenschaften der Nutzerinnen und Nutzer des Systems, wenn diese z.B. abgemeldet oder schlichtweg über Jahre inaktiv sind. Die Gründe werden in den folgenden zwei Unterkapiteln genau diskutiert.

3.1.1 Dunkle Daten durch fehlende Metadaten

Dunkle Daten entstehen durch fehlende Anreize zur Metadatenannotation, mangelnde Integration der Annotation in die Arbeitsprozesse oder das gänzliche Fehlen von Beschreibungsmöglichkeiten wie Metadatenmodellen. In diesem Punkt gibt es eine Übereinstimmung mit der originären Definition von Heidorn, da sie alle Wissenschaftsbereiche einschließt.

Im HPC und damit auch in den computergestützten Ingenieurwissenschaften sind für unterindizierte Daten vor allem fehlende Metadatenmodelle der Grund, aber auch die mangelnde Integration in die Arbeitsprozesse. Es zeigte sich, dass es stark

von der Institutskultur abhängig ist, inwieweit Daten mit Metadaten versehen oder überhaupt auf irgendeine Weise dokumentiert werden. Die konkreten Gründe im betrachteten Fachbereich wurden schon in Kapitel 1.3 diskutiert und sind auf der technischen Seite vor allem in der Vielfalt der Daten und der Nicht-Verfügbarkeit von standardisierten Metadatenmodellen zu suchen. Auf der Seite der Organisation ist es das Fehlen von Anreizen sowie fehlende Kenntnisse im Bereich Datenmanagement und Metadatenannotation, die unterindizierte Daten befördern.

Bezogen auf die Wettersimulation wird beispielsweise davon ausgegangen, dass 50 % aller Daten, die über Netzwerke ausgetauscht werden, nicht mit Metadaten versehen sind und somit als dunkle Daten gelten müssen (Heene u. a., 2016). Die Zahl dürfte für die computergestützten Ingenieurwissenschaften weitaus höher liegen, wenn man von technischen Metadaten oder Containerformaten absieht, die die Daten nur unzureichend in der fachwissenschaftlichen Dimension beschreiben. Dunkle Daten durch Unterindizierung lassen sich quantitativ nur sehr schwer erfassen. Grund dafür sind fehlende Metriken, wann eine Datenobjekt als annotiert bzw. dokumentiert gilt[68]. Das Vorhandensein lässt sich nur feststellen, wenn definiert wird, was unter der Dokumentation von Daten verstanden wird. Hier können die FAIR-Prinzipien[69] eine Metrik anleiten. Die Daten müssen mit reichhaltigen Metadaten beschrieben sein und in einer suchbaren Ressource registriert und indiziert sein. Außerdem müssen die Daten ausführlich mit einer Vielzahl an Attributen beschrieben sein, deren Herkunft aufgezeichnet sowie domänenspezifische Standards mit einbezogen werden. Damit lässt sich prüfen, ob Forschungsdaten dunkel sind und welche dazu zählen. Dieser Metrik folgend, ist der große Teil der ingenieurwissenschaftlichen Daten, die im Rahmen der Dissertation analysiert wurden, dunkel. Dies ist der Fall weil die wenigsten Daten in einer suchbaren Ressource registriert oder mit Metadaten versehen sind, wie sich durch die Analyse und Abschätzung ergab[70].

Eine quantitative Erfassung wird durch die Tatsache erschwert, dass man zur automatisierten Überprüfung nach dunklen Daten den kompletten Datenbestand durchsuchen müsste, also Millionen Datenobjekte. Da sich viele der Daten auf Bandarchivsystemen befinden und diese zur Überprüfung vom Band zurückgeholt werden müssten (sofern die Metadaten nicht in einer Datenbank, sondern auf Band gespeichert sind), wird der Zeitaufwand drastisch erhöht und wichtige Ressourcen werden blockiert. Neben der automatisierten Überprüfung könnte eine Suche nach dunklen Daten über Befragungen erfolgen, jedoch kann diese Methode aus anderen Gründen zu Problemen führen, wie z.B. mangelndes Interesse oder Verzerrungseffekte.

[68] Gilt eine natürlichsprachliche Beschreibung in einer Datei, die im selben Ordner liegt, schon als Metadatum? Oder Informationen in einem Wiki?

[69] Siehe Kapitel 2.3.1.

[70] Durch das Projekt DIPL-ING hatte der Autor Einblick in das Dateninventar und -management vieler Fachbereiche der Ingenieurwissenschaften.

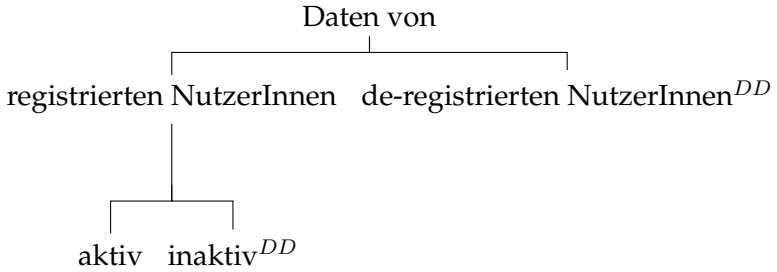

Abbildung 3.3: Typisches Nutzeruniversum (bezogen auf den Status der Accounts) in Rechenzentren. Nutzerinnen und Nutzer, die potentiell dunkle Daten durch ihre Nicht-Aktivität hinterlassen könnten, sind mit einem hochgestellten ^{DD} markiert. Die Grafik wurde vorab publiziert in (Schembera und Durán, 2019).

3.1.2 Dunkle Daten durch Inaktivität

Zweitens entstehen dunkle Daten – im Gegensatz zu Heidorns Idealisierung – auch auf festen Speichersystemen, die nicht, wie ein USB-Stick, verlorengehen können. Nutzerinnen und Nutzer von HPC-Systemen speichern ihre Daten nicht auf portablen Datenträgern, sondern zunächst auf parallelen Dateisystemen und später in zentralen Bandarchivsystemen[71], die einen festen Standort in einem Rechenzentrum haben. Diese Datenträger können nicht, wie Heidorn es metaphorisch ausdrückt, in der Schublade verschwinden. Allerdings können hier Daten auch insofern dunkel werden, dass je nach Verhalten der Nutzerinnen und Nutzer bzw. der Administratorinnen und Administratoren Daten auf dem Speichersystem nicht mehr zugreifbar sind. Das geschieht z.B. durch einen geschlossenen Account, der Daten hinterlässt, deren Herkunft und Weiterbestehen nicht klar durch Verträge geregelt war oder durch eine oder einen NutzerIn, die/der zwar einen Account besitzt, aber nicht mehr auf diesen zugreift. Diese Situation tritt typischerweise ein, wenn Nutzerinnen oder Nutzer ihr Rechenprojekt beendet haben oder beispielsweise Doktorandinnen oder Doktoranden ein Institut verlassen und dabei ihre Daten auf den Systemen zurückbleiben. Nutzerinnen und Nutzer im HPC lassen sich also in zwei Kategorien einteilen, nämlich de-registrierte und registrierte, was in Abbildung 3.3 dargestellt ist. Die *de-registrierten Nutzerinnen und Nutzer* haben aus verschiedenen Gründen keinen Zugang mehr zum Speichersystem, z.B. weil das Rechenprojekt beendet wurde oder sie gesperrt wurden. Sie haben technisch keinen Zugriff mehr auf die Daten – egal ob sie auf die Daten zugreifen wollen oder nicht. Die *registrierten NutzerInnen und Nutzer*, also diejenigen mit der technischen Möglichkeit des Zugangs, könnten potentiell auf die Daten zugreifen. Allerdings muss hier noch einmal zwischen aktiven und

[71] Dabei können Daten auf verschiedenen Datenträgerarten bzw. -bereichen dunkel werden, so z.B. auf den Home-Verzeichnissen, wo die Nutzerinnen und Nutzer ihre Codes etc. lagern, auf dem parallelen Dateisystem, welches als Arbeitsverzeichnis für die Simulationen gilt oder auf dem Bandarchivsystem. Es spielt für das Konzept keine Rolle, wo die Daten liegen bzw. auf welchen Speichermedien. Von Belang ist nur der Fakt, dass potentiell wertvolle Daten nicht mehr sichtbar sind.

inaktiven Nutzerinnen und Nutzern unterschieden werden. Nutzerinnen und Nutzer können als inaktiv gelten, wenn sie sich für eine gewisse Zeitspanne nicht mehr in das System einloggen und so scheinbar keinerlei Interesse mehr an ihren Daten haben. Dieses Nutzeruniversum ist in 3.3 dargestellt und lässt nun auf potentiell dunkle Daten schließen: Die de-registrierten Nutzerinnen und Nutzer führen unweigerlich zu dunklen Daten – deren Daten sind nicht mehr ohne Administrationsrechte zugreifbar. Die Frage der Verantwortung bleibt damit unklar.

Entstehen dunkle Daten durch Inaktivität, muss diese temporal definiert werden, wozu sich insbesondere die Definition über den letzten Zeitpunkt des Zugriffs eignet. Diese Definition wurde gemeinsam erarbeitet und in (Schembera und Durán, 2019) vorab veröffentlicht. Dabei ist von Bedeutung, um welchen Speicherbereich es sich handelt. Die folgende temporale Definition bezieht sich auf das Bandspeichersystem, also auf Daten, die in einem Archiv zur langfristigen Speicherung vorgesehen sind. Der Zugriff auf einen Account steht maßgeblich für die Aktivität einer Nutzerin oder eines Nutzers. Selbst wenn sie oder er keine Daten transferieren, keine Daten lagern usw. sind sie aktiv, indem sie durch das Einloggen und z.B. ein anschließendes Verzeichnis-Listing einige Informationen über den Status ihrer Daten gewinnen. Eine zeitliche Messung, die den letzten Zeitpunkt des Datentransfers zum Ausgangspunkt nimmt, greift hier zu kurz, weil dadurch nicht der Fall berücksichtigt werden kann, dass Nutzerinnen und Nutzer innerhalb des Verzeichnisses z.B. vorbereitend Daten auf dem System kopieren, aber keine übertragen.

Konkret zu definieren, ab welchem Zeitpunkt Nutzerinnen und Nutzer nun als inaktiv gelten, hängt von mehreren Faktoren ab. Auch wenn die DFG prinzipiell zehn Jahre Speicherung von Daten vorsieht, wurde die Klausel in einer neuen Version abgeschwächt und besagt, dass „Primärdaten, die nicht auf haltbaren und gesicherten Trägern aufbewahrt werden können, in begründeten Fällen verkürzte Aufbewahrungsfristen“ (DFG, 2013, S. 22) besitzen können – eine kürzere Frist stünde also nicht gegen die Empfehlung und ist im HPC durch die großen Datenmengen von hoher Relevanz. Im Bereich HPC kann eine Frist von fünf Jahren Inaktivität angesetzt werden, bis Daten, auf die nicht zugegriffen wird, für potentiell dunkel erklärt werden können. Diese Zahl basiert auf der Lebensdauer von HPC-Systemen, die auf durchschnittlich fünf Jahre veranschlagt wird (Wienke u. a., 2015). Die Lebensdauer ist deshalb von Belang, weil es nach einem Systemwechsel grundsätzlich komplizierter wird, auf die Daten zuzugreifen. Rechner- sowie Speichersysteme ändern sich, ebenso die Zugriffswege, eventuell werden Accounts nicht migriert. Die Festlegung einer Zeitspanne, nach der Daten als dunkel definiert werden, sagt noch nichts über die Güte der Daten, deren mögliche Löschung oder das weitere Verfahren aus. Die Festlegung dient zunächst allein der Bestandserhebung. Weitere Schritte, wie mit den dunklen Daten verfahren werden soll, müssen zwischen den Daten-Eigentümern und der Institution ausgehandelt werden oder vorab in den Nutzungsbedingungen geklärt werden. Die verschiedenen Möglichkeiten, mit dunklen Daten zu verfahren, werden in Kapitel 5.5 in Form von Entscheidungskriterien weiter ausgeführt.

Datum	Daten gesamt	DD durch De-Registrierung	DD durch Inaktivität	DD gesamt	Verhältnis
1.12.2009	270 TB	0	-	0	0 %
1.12.2010	607 TB	1,2 TB	-	1,2 TB	0,2 %
1.12.2011	859 TB	1,65 TB	-	1,65 TB	0,19 %
1.12.2012	1285 TB	3,62 TB	-	3,62 TB	0,28 %
1.12.2013	1763 TB	4,02 TB	-	4,02 TB	0,23 %
1.12.2014	3176 TB	5,22 TB	70,43 TB	75,65 TB	2,38 %
1.12.2015	6259 TB	128,73 TB	59,90 TB	188,63 TB	3,01 %
1.12.2016	11886 TB	225,02 TB	67,69 TB	292,99 TB	2,46 %
1.12.2017	19663 TB	619,03 TB	49,00 TB	668,03 TB	3,40 %
1.12.2018	21031 TB	522,89 TB	58,45 TB	581,35 TB	2,76 %

Tabelle 3.1: Zeitliche Entwicklung (2009-2018) dunkler Daten durch Inaktivität (> 5 Jahre) oder De-Registrierung im Bandspeichersystem des HLRS.

3.2 Erhebung dunkler Daten am HLRS

Im Rahmen der Dissertation wurde die Zahl der dunklen Daten im Bandspeichersystem des HLRS[72] erhoben. Durch das Betriebskonzept des HLRS können dunkle Daten nicht auf dem parallelen Lustre-Dateisystem entstehen, da der Workspace-Mechanismus dies verhindert[73]. Bezogen auf das Bandspeichersystem des HLRS wurden nur die dunklen Daten erfasst, die durch de-registrierte oder inaktive Nutzerinnen und Nutzer entstanden sind. Zur Erhebung wurde von der Account-Aktivität auf die dunklen Daten geschlossen. Dabei wurde zunächst die Zahl der dunklen Accounts festgestellt, wobei Inaktivität der Daten dadurch definiert wurde, dass fünf Jahre oder länger kein Zugriff auf den Account statt fand. Schließlich wurden die Daten, die in den einzelnen dunklen Accounts existieren, addiert. Dafür wurden die Logdateien bis zum 1. Dezember 2018 des Bandarchivsystems, welche die Zugriffe und Datenmengen enthalten, mittels Skripten ausgewertet. Die zeitliche Entwicklung dunkler Daten im Bandspeichersystem des HLRS findet sich in Tabelle 3.1 dargestellt und in Abbildung 3.4 visualisiert. Hierbei zeigt sich, dass die dunklen Daten durch Inaktivität der Nutzerinnen und Nutzer oder durch de-registrierte Accounts nur einen Bruchteil des Gesamtvolumens der gespeicherten Daten umfassen, und sich im unteren Prozentbereich um die 3 % einpendeln. Da das System erst 2009 in Betrieb genommen wurde, sind dunkle Daten durch Inaktivität erst ab 2014 erfasst, weil sich vor diesem Zeitpunkt die Fünfjahresgrenze nicht anwenden lässt. In Abbildung 3.4

[72] Die Speicherhierarchie bzgl. der persistenten Speichermedien am HLRS stellt sich wie folgt dar: Die Simulationsdaten werden auf das parallele Dateisystem *Lustre* geschrieben, welches auf Festplatten basiert. Dort werden die Daten evtl. nachbearbeitet und später entweder zur Heimatinstitution der jeweiligen Nutzerinnen und Nutzer oder in den Auslagerungsspeicher des HLRS verschoben. Der Auslagerungsspeicher ist ein HSM (Hierarchical Storage Management System), welches mit HPSS (High Performance Storage System) betrieben wird. Dieser Auslagerungsspeicher basiert auf Bandspeicher.

[73] Nutzerinnen und Nutzer können sich ein Speicherkontingent, genannt *Workspace* nur für bis zu 30 Tage reservieren. Nach Ablauf der Zeit kann der Workspace verlängert oder ein neuer reserviert werden. Nach Rechenprojektende werden die Workspaces unwiederbringbar gelöscht.

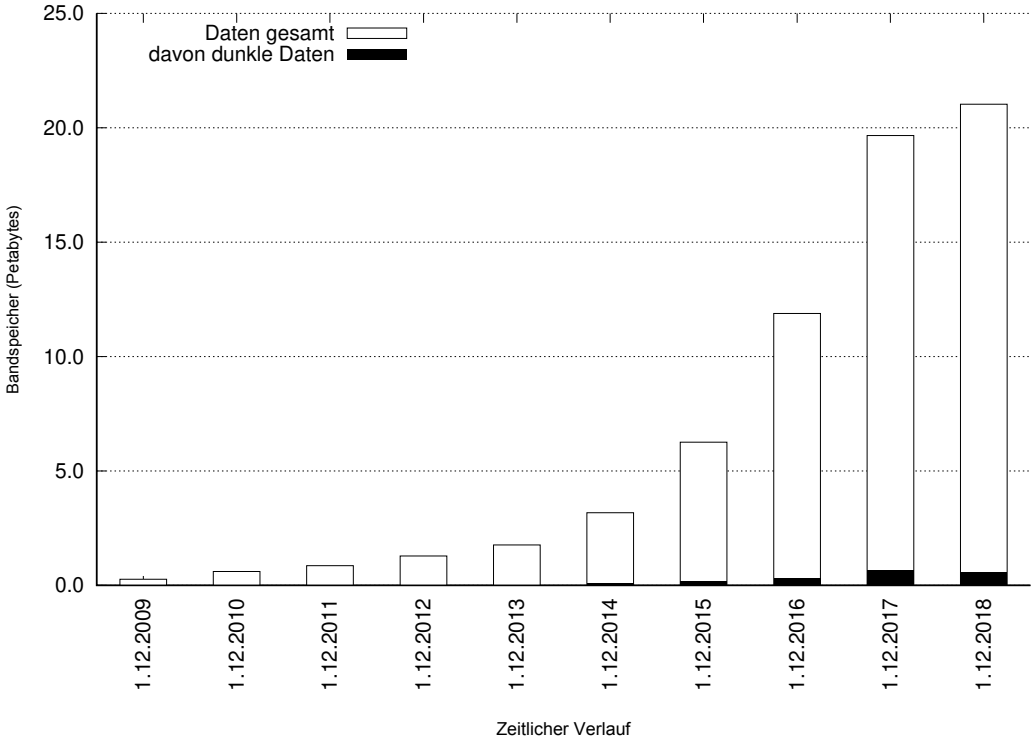

Abbildung 3.4: Zeitliche Entwicklung (2009-2018) dunkler Daten durch Inaktivität (> 5 Jahre) oder De-Registrierung im Bandspeichersystem des HLRS. (DD = Dunkle Daten).

fallen in der gewählten Skalierung die dunklen Daten überhaupt erst ab 2015 auf, da sie hier erstmals über die 1 %-Marke gelangt sind. Die dunklen Daten, die durch inaktive Nutzerinnen und Nutzer erzeugt wurden, machen jeweils den geringeren Teil der gesamten dunklen Daten seit 2015 aus und sind daher in Abbildung 3.4 nicht explizit aufgeführt. In Tabelle 3.1 findet sich die komplette Statistik zu dunklen Daten seit 2009. Der Rückgang der dunklen Daten im Vergleich vom 1.12.2017 zum 1.12.2018 rührt daher, dass Institutsleiterinnen und -leitern Daten von de-registrierten Accounts früherer Mitarbeiterinnen und Mitarbeiter auf eigene Nachfrage hin überantwortet wurden und die Daten so nicht mehr als dunkel markiert sind. Dunkle Daten stellen hier also einen gewissen Bodensatz an Daten da, der sich über die Jahre akkumulieren kann, wenn nicht entsprechende Gegenmaßnahmen, wie sie später in Kapitel 5 diskutiert werden, zum Einsatz kommen.

3.3 Implikationen dunkler Daten

Dunkle Daten können vielfältige ethische und technische, aber auch finanzielle und rechtliche Probleme implizieren. Die Daten, die im System mitgeschleppt werden und weder regelmäßig genutzt (dunkle Daten durch Inaktivität), noch zugänglich (dunkle Daten durch De-Registrierung) oder verstehbar (dunkle Daten durch fehlende Metadatenannotation) sind, können der Grund vieler Probleme sein, die im Folgenden besprochen werden.

3.3.1 Ressourcenökonomische Implikationen

Da Datenhaltung insbesondere finanzielle Ressourcen kostet, können dunkle Daten Ressourcenverschwendung bedeuten. Die Kosten für die Speicherung der Daten umfasst nicht nur die Kosten der Medien, sondern die Gesamtkosten des Betriebs (Total Cost of Ownership, TCO), also auch die Laufwerke, die Bibliothek sowie Stromkosten usw. Beispielsweise kostet, der Studie (*Quantum White Paper. LTO: The New "Enterprise Tape Drive"* 2018) folgend, die Datenhaltung von 5 PB bei einer durchschnittlichen Schreibrate von 10 TB/Stunde je nach Technologie zwischen 302.550$ und 627.250$. Geht man nur von 3 % dunklen Daten aus, würden in diesem konkreten Setup also zwischen 9.076$ und 18.817,5$ für dunkle Daten verschwendet. Typischerweise besitzen Höchstleistungsrechenzentren bzw. Rechenzentren im Allgemeinen aber mindestens zwei Kopien der Daten, womit sich auch die Zahl der Kosten und damit die Zahl der verschwendeten Ressourcen verdoppelt. In die TCO gehen allerdings weder die Kosten der Erzeugung der Daten selbst noch die Ausbildungs- und Personalkosten ein. Alle diese Kosten gehen nur vermittelt in die Daten ein. Sind Daten dunkel, sind die Ressourcen verschwendet, da die Daten nicht mehr nachgenutzt sondern neu produziert werden müssen und Medien durch eine unklare Situation potentiell blockiert werden.

3.3.2 Rechtliche Implikationen

Auch rechtlich können dunkle Daten relevant werden, insbesondere wenn sie personenbezogene Daten beinhalten. Dies gewinnt besonders bei dunklen Daten an Bedeutung, die durch de-registrierte Nutzer entstehen, da hier der Account abgemeldet ist. Die DGSVO regelt in Artikel 17[74] klar, dass Daten mit Personenbezug sofort zu löschen sind, wenn diese für „die Zwecke, für die sie erhoben oder auf sonstige Weise verarbeitet wurden, nicht mehr notwendig" (DSGVO, §17, Abs 1) sind. Simulationsdaten im Höchstleistungsrechnen beinhalten jedoch nur sehr selten personenbezogene Daten, allerdings kann dies in anderen Fachbereichen oder Rechenzentren durchaus von Bedeutung sein.

Ebenso sind Eigentumsrechte an dunklen Daten im Rahmen des Urheberrechts zu thematisieren. Dabei gelten nur Daten von einer gewissen Schöpfungshöhe (Individualität und Kreativität muss erkennbar sein) als vom Urheberrecht geschützt. Hierzu muss geprüft werden, ob die Forschungsdaten daunter fallen (Lauber-Rönsberg, Krahn und Baumann, 2018). Sollten Urheberrechte greifen und die Daten als dunkel bestimmt worden sein, ist der weitere Verfahrensweg unklar, da die Daten beispielsweise nicht einfach gelöscht werden, aber evtl. auch nicht öffentlich zugänglich gemacht werden können.

[74] `https://www.datenschutz-grundverordnung.eu/grundverordnung/art-17-ds-gvo/`, Zugriff 2.3.2019.

3.3.3 Ethische Implikationen

Dunkle Daten zeigen Erkenntnisprobleme, da Daten ohne Metadatenauszeichnung oder einen Eintrag in einer zentralen Datenbank nicht wahrgenommen werden können. Dies kann wiederum rechtliche Implikationen verursachen, wenn diese vergessenen Daten personenbezogene Informationen enthalten. Selbst, wenn die Existenz dunkler Daten festgestellt wurde, ist damit u.U. noch nicht geklärt, an welchen Speicherorten sie überall existieren und wie sie interpretierbar sind.

Darüber hinaus kann es zu Verantwortungsproblemen durch dunkle Daten kommen. Sind dunkle Daten durch de-registrierte Nutzerinnen und Nutzer festgestellt und lokalisiert, ist bei unzureichender Datenbeschreibung trotzdem nicht klar, wem diese nicht mehr zugeordneten dunklen Daten gehören und wie damit weiter verfahren werden sollte. Es ist ein Problem der Verantwortung, ob und wann dunkle Daten gelöscht werden können, wenn diese nicht mehr zugeordnet sind. Dabei ist insbesondere der Fall, dass dunkle Daten von inaktiven Nutzerinnen und Nutzern herrühren kritisch. Es besteht zwar noch ein Account und auch eine technische Möglichkeit des Zugriffs, allerdings scheint die mit dem Account assoziierte Person kein Interesse mehr zu haben. Fehlt darüber hinaus Metadatenannotation, ist auch nicht klar, wie die Daten verstanden werden können und welche Nutzen sie überhaupt noch haben.

3.3.4 Dunkle Daten sind nicht FAIR

Dunkle Daten stehen FAIRen Daten diametral entgegen. Im Folgenden wird gezeigt, dass dunkle Daten keines der Merkmale von FAIRen Daten erfüllen. Die FAIR-Prinzipien finden sich in Kapitel 2.3.1 detailliert dargestellt und werden hier im Einzelnen gegen die jeweiligen Arten der dunklen Daten geprüft, wobei die Kennung in den Klammern jeweils für das Kürzel der jeweiligen FAIR-Forderung steht. So steht *(F1)* beispielsweise für die Forderung *Findability 1: Reichhatlige Metadatenbeschreibung*.

Soll die Findbarkeit der Daten sichergestellt werden, müssen diese gemäß FAIR-Prinzipien mit reichhaltigen Metadaten (F1) und einer persistenten Kennung (F2) – innerhalb der Metadaten (F4) – beschrieben und in einer auffindbaren bzw. durchsuchbaren Ressource registriert sein (F3). Dunkle Daten durch fehlende Metadatenannotation entziehen sich der Findbarkeit, indem sie nicht mit einer eindeutigen Kennung bzw. Metadaten versehen sind und ihre Existenz oft unklar ist. Dunkle Daten durch De-Registrierung entziehen sich der Findbarkeit, indem die Daten in einer suchbaren Ressource nicht mehr mit einem Account registriert sind. Dunkle Daten durch Inaktivität können prinzipiell alle Forderungen bzgl. Findbarkeit erfüllen, allerdings ist das Interesse, sie insbesondere nachträglich findbar zu machen gering und die Verfügbarkeit der Erstellerin oder des Erstellers unklar. Inaktivität der Nutzerinnern und Nutzer zeichnet sich gerade durch ihre Nichtverfügbarkeit (und evtl. auch Nichterreichbarkeit oder Desinteresse) aus.

	DD durch fehlende MD	DD durch De-Registrierung	DD durch Inaktivität
F1	–	✓	–
F2	–	✓	–
F3	–	–	–
F4	–	✓	–
A1	–	–	–
A2	–	–	✓
I1	–	–	✓
I2	–	–	✓
I3	–	–	✓
R1	–	✓	✓

Tabelle 3.2: Übereinstimmung der Arten dunkler Daten mit den jeweiligen Dimensionen der FAIR-Prinzipien. (Legende: MD = Metadaten, DD = dunkle Daten, ✓ = Übereinstimmung, – = Nicht-Übereinstimmung)

Zugreifbarkeit bedeutet im Sinne der FAIR-Prinzipien, dass Daten mittels Standardprotokollen und ihrer PID zugänglich sein müssen (A1), mindestens aber ihre Metadaten auch bei Datenunverfügbarkeit sichtbar sein müssen (A2). A1 kann nicht erfüllt werden, wenn die Daten durch de-registrierte Nutzerinnen und Nutzer entstehen, da diese dunkle, d.h. verwaiste, Accounts hinterlassen. A2 kann nicht erfüllt werden, wenn die Daten durch fehlende Metadatenannotation dunkel werden. Bei dunklen Daten durch Inaktivität sind die Daten prinzipiell noch zugreifbar.

Die Interoperabilität gemäß den FAIR-Prinzipien bedeutet, dass Metadatenbeschreibung durch formale und verfügbare Sprachen erfolgen muss (I1), außerdem, dass sowohl Daten als auch Metadaten Vokabulare nutzen, die den FAIR-Prinzipien folgend (I2) Referenzen zu anderen Daten und Metadaten enthalten (I3). Sind dunkle Daten durch fehlende Metadatenannotation entstanden, so wird keine der Forderungen I1-I3 erfüllt. Dunkle Daten durch De-Registrierung oder Inaktivität sind hier nicht betroffen, da nur Forderungen bzgl. der Metadaten gestellt werden.

Die Nachnutzbarkeit von Forschungsdaten wird insbesondere darüber erreicht, dass Daten und Metadaten mit einer Vielzahl von genauen und relevanten Attributen beschrieben werden (R1). Dunkle Daten durch fehlende Metadaten widersprechen dieser Forderung komplett. De-Registrierte und inaktive dunkle Daten können, sofern sie mit Metadaten annotiert sind, diese Forderung erfüllen.

Tabelle 3.2 zeigt eine Zusammenfassung des Abgleichs der Arten dunkler Daten im HPC mit den Forderungen der FAIR-Prinzipien. Dabei erfolgt die Betrachtung in der Tabelle nach der Idealisierung, dass sich die Gründe für die dunklen Daten nicht überschneiden, d.h. dunkle Daten nur entweder durch fehlende Metadaten, durch De-Registrierung oder durch Inaktivität entstehen, aber nicht aus mehreren

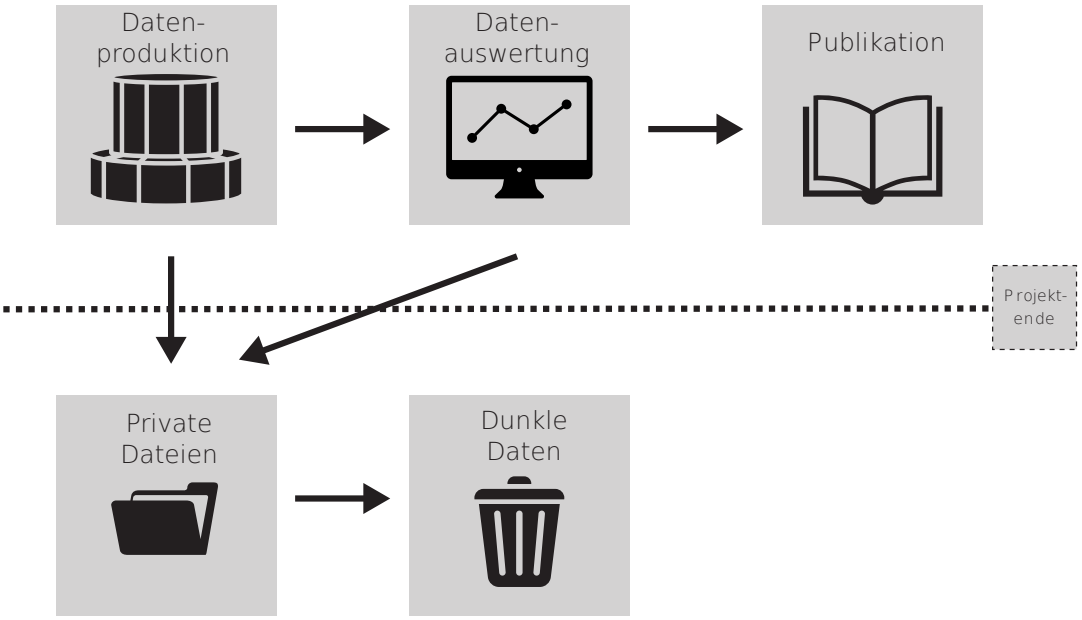

Abbildung 3.5: Heutiger Datenlebenszyklus im HPC. Nach Produktion und Auswertung der Daten und darauf folgender Publikation der Resultate werden die Daten in einen privaten Bereich verschoben, wo sie gelöscht oder vergessen werden. Vorab veröffentlicht in (Schembera und Bönisch, 2017).

Gründen gleichzeitig[75]. Dunkle Daten durch De-Registrierung oder Inaktivität bringen nur Probleme bzgl. der FAIR-Prinzipien, wenn dies die Zugreifbarkeit und die Verfügbarkeit einschränkt[76].

Wie gezeigt wurde, stehen sich dunkle Daten und die Idee der FAIRen Daten antagonistisch gegenüber. Dunkle Daten, insbesondere die durch fehlende Metadatenannotation erzeugten, widersprechen in allen Punkten den Forderungen der FAIR-Prinzipien.

3.3.5 Dunkle Daten sind dem Datenlebenszyklus inhärent

Der gegenwärtige Datenlebenszyklus im HPC, der sich an den Phasen Datenproduktion, Datenauswertung und Publikation orientiert[77], ist in Abbildung 3.5 dargestellt. Die Daten entstehen durch die Datenproduktion auf Großrechenanlagen wie der CRAY XC40 am HLRS und werden schließlich in jenen zentralen Einrichtungen, auf institutseigenen Clustern oder PCs ausgewertet. Darauf folgt die Publikation. Während dieser Zeit sind die Daten aktiv und werden generiert, bearbeitet, ausgewertet und zwischengespeichert. Das Projektende markiert meist den Endpunkt im Datenlebenszyklus. Die Daten sind für die einzelnen Wissenschaftlerinnen und

[75] Obwohl diese Annahme in der Praxis selten zutrifft und sich dunkle Daten meist aus mehreren Gründen gleichzeitig ergeben, wurde die Annahme aus analytischen Gründen getroffen, um Nichtübereinstimmung von dunklen Daten in den jeweiligen Kategorien genau zu explizieren.

[76] Die FAIR-Prinzipien setzen Metadaten als oberstes Handlungsideal, um das Management von Daten zu verbessern. Fragen nach der Verantwortung und nach der Verfügbarkeit von Daten werden von den FAIR-Prinzipien nicht ausreichend behandelt.

[77] Siehe Kapitel 1.2.2.

Wissenschaftler nicht mehr von Bedeutung, da die Ergebnisse in Form der wissenschaftlichen Publikation in Bibliotheken permanent fortbestehen und die Publikation einzig für die Reputation von Wissenschaftlerinnen und Wissenschaftlern von Belang ist. Da die Daten nach Projektende in private Verzeichnisse verschoben werden und/oder nicht mit Metadaten versehen wurden, sind dunkle Daten dem gegenwärtigen Datenlebenszyklus in den meisten Disziplinen und vor dem Hintergrund der in Kapitel 1.3 Kapitel herausgearbeiteten Probleme des heutigen Datenmanagements inhärent.

3.3.6 Forschungsdatenmanagement muss dunkle Daten reduzieren

Wie sich in den vorigen Abschnitten zeigte, haben dunkle Daten rechtliche, ethische, technische und finanzielle Probleme zur Folge. Um diese erst gar nicht aufkommen zu lassen, muss Forschungsdatenmanagement zuallererst dunkle Daten vermeiden bzw. vollständig reduzieren. Bezogen auf dunkle Daten durch fehlende Metadatenannotation muss deren Reduzierung insbesondere auf die Herstellung FAIRer Daten hinarbeiten, da diese Metadatenannotation voraussetzt. Dunkle Daten durch De-Registrierung und Inaktivität müssen durch andere, insbesondere organisatorische Verfahren vermindert werden. Diese Ziele müssen auf mehreren Ebenen bearbeitet werden und werden in Kapitel 4 näher definiert. In Kapitel 5 wird mit konkreten Lösungsansätzen versucht, die Reduzierung dunkler Daten zu erreichen. Die große Herausforderung dabei ist die fehlende Metadatenannotation, was zugleich den zentralen Angriffspunkt darstellt, der in Kapitel 5.2 aufgegriffen wird.

Kapitel 4

Anforderungen an Forschungsdatenmanagement für HPC

Wurden in Kapitel 1.3 die Herausforderungen von Forschungsdatenmanagement im HPC bzw. in der computergestützten Modellierung herausgearbeitet und in Kapitel 3 die daraus resultierenden dunklen Daten problematisiert, sollen daraus nun in diesem Kapitel Anforderungen abgeleitet werden, die später genutzt werden, um das Systemdesign und das Konzept zu begründen. Die Anforderungen, die in diesem Kapitel aufgestellt werden, ergeben sich meist direkt aus den Herausforderungen in Kapitel 1.3 oder aus den Vorarbeiten in Kapitel 2. Wurde in Kapitel 1.3 hergeleitet, dass es ein Problem gibt, und in Kapitel 3 gezeigt, dass es in dunklen Daten kulminiert, wird in diesem Kapitel definiert, was FDM zur Vermeidung von dunklen Daten genau lösen muss. Die Anforderungen in diesem Kapitel sollen also genau auf die Frage Antwort geben, *was* gelöst werden muss, um FDM im HPC zu ermöglichen und dunkle Daten zu vermeiden. Der Frage, *wie* es gelöst wird, wird dann in Kapitel 5 nachgegangen und mit einem Konzept bzw. einer Strategie und einer Implementierung begegnet. Im Folgenden wird die Zweiteilung in einen *technischen* und *organisatorischen* Teil beibehalten, die in den Herausforderungen unternommen wurde. Die technischen Anforderungen leiten sich aus den technischen Herausforderungen ab, die organisatorischen Anforderungen aus den entsprechenden organisatorischen. Die Anforderungen werden dabei möglichst auf die zwei Anwendungsfälle (Daten aus der Thermodynamik und aus der Aerodynamik) bezogen, sind jedoch teilweise auch allgemein für Forschungsdatenmanagement im HPC oder generell gültig.

4.1 Technische Anforderungen

4.1.1 Metadaten

Metadaten sind eine der wesentlichen Anforderungen für gelingendes Forschungsdatenmanagement, da nur sie interpersonelle Verständlichkeit garantieren:

> „Das Forschungsdatenmanagement muss so gestaltet werden, dass Datenzugriff und -auswertung unabhängig vom Datenerzeuger möglich wird und bleibt. Neben der technischen Speicherung und Lesbarkeit der Forschungsdaten müssen ausreichend Informationen zu ihrer Interpretation in Metadaten überliefert werden.“ (Büttner, Hobohm und Müller, 2011, S. 14)

Metadaten sind das Hauptkonzept für gelingendes Forschungsdatenmanagement (Gray u. a., 2005; Potthoff u. a., 2014) und somit für die Vermeidung dunkler Daten. Wie in Kapitel 3 gezeigt wurde, sind fehlende Metadaten die Hauptquelle für dunkle Daten. Metadaten sind somit für das Vorhaben der Dissertation, nämlich der Optimierung von Forschungsdatenmanagement und der Reduzierung dunkler Daten in den computergestützten Ingenieurwissenschaften, die wesentliche Anforderung.

Metadaten haben gemäß den FAIR-Prinzipien, die in Kapitel 2.3.1 detailliert besprochen wurden, mehrere Aufgaben (Wilkinson u. a., 2016): Sie sollen die Daten findbar machen, indem sie die Daten mit einer reichhaltigen Beschreibung und einer persistenten Kennung ausstatten. Diese fungiert als Zeiger auf die Daten, der auch bei der Bewegung der Daten bestehen bleibt und den richtigen Ort der Daten anzeigt. Um die Findbarkeit der Daten im konkreten Anwendungsfall der Ingenieurwissenschaften sicherzustellen, müssen Metadaten als potentielle Suchkriterien bestimmt werden. Metadaten sollen die Daten insbesondere interoperabel und nachnutzbar machen. Dies wird durch eine formale Beschreibungssprache für die Daten gewährleistet sowie durch eine Beschreibung der Daten durch präzise und relevante Attribute, die auch ihre Herkunft und domänenspezifische Informationen enthalten. Um die Daten also zu verstehen und damit nachnutzen zu können, müssen alle für die computergestützten Ingenieurwissenschaften relevanten Informationen (z. B. über die genutzte Software oder die Rechenumgebung) in den Metadaten abgelegt sein. Dies ist fundamental wichtig für die Simulationswissenschaften im Allgemeinen und die Ingenieurwissenschaften im Speziellen, da nicht von strenger Reproduzierbarkeit der Daten ausgegangen werden kann[78]. Dieser Anforderung wird durch die siebte DFG-Empfehlung zur guten wissenschaftlichen Praxis speziell für die numerische Simulation noch Nachdruck verliehen:

> „Experimente und numerische Rechnungen können nur reproduziert werden, wenn alle wichtigen Schritte nachvollziehbar sind. Dafür müssen sie aufgezeichnet werden. [...] dass die Arbeiten an anderem Ort nachvollzogen werden können.“ (DFG, 2013, 21f)

[78] Zur Reproduzierbarkeit siehe Kapitel 1.2.

Die Anforderungen an ein Metadatenschema für die computergestützten Ingenieurwissenschaften können also nur erfüllt werden, wenn der Datensatz nicht nur technisch durch *technische Metadaten* und allgemein durch *deskriptive Metadaten* beschrieben wird. Es müssen darüber hinaus auch das betrachtete System mittels *fachspezifischer Metadaten* beschrieben werden sowie der Weg der Entstehung als *Prozessmetadaten* mit in die Anforderungen eingehen. Diese Prozessmetadaten beschreiben den konkreten Erzeugungsprozess der Daten (in der Thermodynamik z.B.: Equilibrierung, Simulationslauf, Auswertung; generell aber auch z.B. die Rechnerumgebung). Die fachspezifischen Metadaten beschreiben, was genau simuliert wurde. Die Einteilung in die vier Metadatenkategorien *technische Metadaten*, *deskriptive Metadaten*, *Prozessmetadaten* und *fachspezifische Metadaten* wird im weiteren Verlauf der Arbeit beibehalten[79]. Die konkreten Anforderungen an die Metadaten (also *welche* Informationen genau gespeichert werden sollen) speisen sich in der vorliegenden Untersuchung aus den konkreten Anforderungen der Fachbereiche, im Fall der vorliegenden Dissertation der Thermo- und der Aerodynamik. Sie resultieren im konkreten Metadatenmodell *EngMeta*, dessen Entwicklung in Kapitel 5.2 entlang der vier Kategorien vollzogen wird.

4.1.2 Integration in den Arbeitsprozess

Wie sich in den Kapiteln 1.3.2, 2.3.1 und 3 zeigte, ist die Dokumentation der Daten durch Metadaten der wesentliche Punkt bei der Reduzierung der dunklen Daten und für die Förderung von FAIRen Daten. Dokumentation durch Metadaten wird im Forschungsalltag allerdings nur durchzusetzen sein, wenn die Metadatendokumentation während des Forschungsprozesses möglichst automatisiert abläuft, um die Hemmschwelle durch den zusätzlichen Arbeitsaufwand möglichst gering zu halten. Da sich einige der zu dokumentierenden Informationen selten ändern (z.B. die Kontaktinformationen) oder wiederholen (wie Projekteigenschaften), muss es möglich sein, solche Informationen in Profilen ablegen zu können. Andere Metadaten wiederum müssen automatisiert erfasst oder erzeugt werden können, z.B. Informationen aus den POSIX-Dateieigenschaften heraus (wie Dateigröße, -name, Prüfsummen) oder fachspezifische Informationen, die aus den Logdateien der Simulationscodes extrahierbar sind. Zwar ist anfänglich ein Entwicklungsaufwand nötig, allerdings sollte die Unterstützung zur automatisierten Metadatenextraktion möglichst generisch entwickelt werden. Diese muss so in den Workflow integriert werden, dass die Software im Workspace eines HPCs lokal ohne Administratorprivilegien lauffähig ist. Ebenso muss gewährleistet werden, dass keine oder so wenig Zusatz-Software wie möglich installiert werden muss, weder aufseiten der Nutzerinnen und Nutzer des HPC-Systems noch auf der Seite der Administratorinnen und Administratoren.

[79] Die Einteilung entspricht im Wesentlichen der Trennung, die das OAIS-Modell vorschlägt (siehe Kapitel 2.3.2), wonach Metadaten in deskriptive, administrative und strukturelle Metadaten kategorisiert werden können, die um einen fachspezifischen Teil ergänzt wird.

In Kapitel 5.3 wird dargestellt, wie eine automatisierte Metadatenerfassung generisch implementiert werden kann. Eine weitere Anforderung an eine automatisierte Metadatenextraktion ist eine Laufzeit, die im Verhältnis zur Ausführungszeit der jeweiligen Simulation stehen muss. Für typische Simulationsläufe auf Höchstleistungsrechnern sollte für die Metadatenextraktion eine Zeit von wenigen Sekunden bis Minuten veranschlagt werden.

4.1.3 Nutzerschnittstelle

Die Nutzerschnittstelle muss sich in den HPC-Workflow integrieren lassen. Die Nutzerschnittstelle muss Möglichkeiten bereitstellen, Daten einzupflegen, zu suchen, zu manipulieren und zu browsen. Ebenso müssen Möglichkeiten bestehen, die Daten mit Metadaten zu annotieren. Wichtig dabei ist, dass Ortstransparenz gefordert sein muss, da der tatsächliche Speicherort der Daten vom Zugriffsort aus Gründen der Skalierbarkeit abweichen kann. Die Nutzeroberfläche mit ihren wesentlichen Funktionalitäten von den HPC-Frontend-Knoten aus einem lokalen Nutzerverzeichnis zugreifbar sein. Dabei ist eine Schnittstelle als Kommandozeilenprogramm von hoher Priorität, da der typische Arbeitsprozess im HPC auf diese Weise erfolgt und grafische Tools von nachrangiger Bedeutung sind. Dies gilt zumindest für alle FDM-Prozesse, die nah an der Erzeugung der Daten stattfinden, wie z. B. die Metadatenannotation. Eine grafische Benutzeroberfläche, z. B. in Form einer Weboberfläche, ist für die Kuration sowie für das Browsen der Daten allerdings ungemein hilfreich. Die Nutzerschnittstelle kann also in zwei Teile aufgeteilt werden. Zum einen ist dies eine Nutzerschnittstelle, die mittels Kommandozeile zugreifbar ist, um auf oder von dem HPC-System ausgehend FDM-Operationen durchzuführen. Zum anderen ist dies eine grafische Benutzeroberfläche, um von außerhalb der HPC-Umgebung auf das FDM-System zugreifen zu können. Auf der automatisierten Metadatenerfassung aufbauend, die in Kapitel 5.3 vorgestellt wird, wird eine sich nahtlos in den Forschungsprozess integrierbare Lösung vorgestellt, die die Daten nach der Erfassung direkt in ein Repositorium spielt.

4.1.4 Skalierbarkeit

Um der Herausforderung der großen Datenmengen, wie sie für simulationsgestützte Ingenieurwissenschaften typisch ist[80], gerecht zu werden, müssen Systeme und Speichermethoden entsprechend skalierbar sein. Dies bedeutet, dass das System auch für Datenobjekte in der Größenordnung von Terabytes effizient durchsuchbar sein und auch deren Rückholbarkeit sicherstellen muss. In Kapitel 5.1 wird gezeigt, wie die Skalierbarkeit durch ein spezifisches Systemdesign sichergestellt werden kann.

[80] Siehe Kapitel 1.3.1

4.1.5 Datensicherheit und -integrität

Die Datensicherheit der gespeicherten Forschungsdaten ist eine Anforderung, die im Bereich der computergestützten Simulation einigen Spezifika unterliegt und insbesondere von den Herausforderungen der Datengröße flankiert wird. Jegliche Überlegungen zur Datensicherheit müssen unter diesem Aspekt geprüft werden. Zwar schreibt die DFG prinzipiell zehn Jahre Speicherung von Daten vor, allerdings wurde die Klausel in einer neuen Version abgeschwächt und besagt nun, dass „Primärdaten, die nicht auf haltbaren und gesicherten Trägern aufbewahrt werden können, in begründeten Fällen verkürzte Aufbewahrungsfristen“ (DFG, 2013, S. 22) besitzen können. Demnach muss ein FDM-System eine gewisse Aufbewahrungsfrist vorsehen. Die Dauer der Frist unterliegt aber einem konkreten, nicht technischen Aushandlungsprozess. Es muss außerdem möglich sein, je nach Wichtigkeit der Daten, unterschiedlich viele Kopien anlegen zu können. Zwei Kopien, idealerweise an räumlich unterschiedenen Orten, gelten derzeit als obligatorisch für eine gute FDM-Praxis, die Bitstream Preservation ermöglicht (Potthoff u. a., 2014). Ebenso muss die Unveränderlichkeit der Daten überprüfbar sein, was nur über die Nutzung von Checksummen möglich ist. Das FDM-System muss beim Einspielen der Daten eine Checksumme bilden und diese mit im Metadatensatz speichern können. Es sollte auch die Möglichkeit von periodischer Überprüfung der Checksummen vorhanden sein. Außerdem sollte das System Ende-zu-Ende-Datenintegrität erlauben. Dies bedeutet, dass die Checksummen vor und nach dem Einspielen miteinander abgeglichen werden. Der Datentransfer gilt nur dann als erfolgreich, wenn beide übereinstimmen (Hick, 2013). Verschlüsselung spielt im HPC eine untergeordnete Rolle, da die Forschung, die zu den Daten führt, meist öffentlich finanziert ist und keine personenbezogenen oder streng vertraulichen Daten produziert. Außerdem stehen den Vorteilen der Verschlüsselung der Aufwand (zeitlich gesehen) für die Verschlüsselung von Petabytes von Daten gegenüber. Grundsätzlich sollte aber die Möglichkeit vorgesehen werden, zumindest einzelne Datensätze verschlüsseln zu können.

4.2 Organisatorische Anforderungen

4.2.1 Anreize

Forschungsdatenmanagement bedeutet zunächst Mehraufwand durch die Datendokumentation und Aufbereitung der Daten, weswegen die Frage berechtigt ist, warum genau Anstrengungen in diese Richtung unternommen werden sollten. Es gibt jedoch intrinsische und extrinsische Gründe, warum Wissenschaftlerinnen und Wissenschaftler bzw. Institutionen Forschungsdatenmangagement vorantreiben sollten. Extrinsische Gründe sind externe Faktoren wie Regularien oder Standards, wie z.B. die DFG-Richtlinie. Intrinsische Gründe zielen auf die Beteiligten selbst, indem z.B. Forschungsdatenmanagement zu einer höheren Reputation führt. Anreize dieser

Art müssen geschaffen und verfestigt werden, um Forschungsdatenmanagement zu etablieren. Dabei gehen die Anreize vermittelt in andere Anforderungen ein. Intrinsische Anreize sind vor allem die möglichst nahtlose Integration in den Arbeitsprozess sowie das entsprechende Design der Nutzerschnittstelle. Extrinsische Anreize lassen sich wesentlich schlechter implementieren, da sie die Unterstützung von höheren Instanzen benötigen. Daher muss eine Instanz, deren Aufgabe die Unterstützung und Durchsetzung von Forschungsdatenmanagementprozessen ist, zur Anforderung erhoben werden.

4.2.2 Datenmanagementpläne

Datenmanagementpläne (DMP) sind formale Dokumente, die Zahlen und Planungen bzgl. des Forschungsdatenmanagements festlegen. Sie gelten im Sinne der Organisationssicherheit als Anforderung an gutes FDM (Jensen, 2011). Die Dokumente legen fest, wie mit den Daten während und nach dem Forschungsprojekt umgegangen wird und stellen eine Aushandlung zwischen den verschiedenen Parteien dar. Für die erfolgreiche Einreichung von Forschungsanträgen an die Europäische Union oder an die National Science Foundation sind DMPs heute schon Pflicht (EU, 2016b; NSF, 2014). Die DFG sieht diese auch als unabdingbare Voraussetzung für Nachnutzbarkeit von Forschungsergebnissen (DFG, 2018).

Ein DMP muss auch für Forschungsdatenmanagement im HPC zu einer Anforderung für Rechenprojekte erhoben werden, da nur aufgrund einer Aushandlung zwischen den Datenproduzenten, den Betreibern des HPC-Systems sowie der Speichereinrichtung Datenmanagement betrieben werden kann. Dies muss in einem verbindlichen Dokument festgehalten werden. DMPs tragen wesentlich zur Reduzierung dunkler Daten bei, die durch de-registierte oder inaktive Nutzerinnen und Nutzer zustande kommen. Ist durch den DMP geregelt, wie mit hinterlassenen Daten verfahren wird, wenn ein Account abgemeldet wurde, können dunkle Daten maßgebend reduziert werden.

4.2.3 Qualifikation und Ausbildung

Für nationale und internationale Forschungsförderer (BMBF, DFG, OECD, ARC, Wellcome Trust) ist der Studie (Wilms u. a., 2018) zufolge Ausbildung im Bereich FDM mittlerweile eine wichtige Anforderung. Nur indem Schulungsaktivitäten intensiviert werden, können Wissenschaftlerinnen und Wissenschaftler von FDM profitieren. Qualifikation und Ausbildung wird auch in den erweiterten FAIR-Prinzipien als Handlungsempfehlung genannt. Gemäß (Calhoun u. a., 2016) dient Training und Qualifikation vor allem dazu, aus der Technologie entspringende Unannehmlichkeiten abzupuffern oder zu vermeiden, die bei Großgeräten wie Höchstleistungsrechnertechnologie stärker ausfallen als in kleineren Skalen.

Qualifikationsmaßnahmen müssen sowohl speziell geschultes Personal betreffen als auch die Wissenschaftlerinnen und Wissenschaftler selbst zu einem gewissen

Grade (Mattmann, 2013). Zur Qualifikation müssen Kurse und Schulungen abgehalten werden. Außerdem ist eine Integration ins Curriculum denkbar, die schon Studierende an Themen des Datenmanagements heranführt.

4.2.4 Ein spezifischer Datenkurator

Bezogen auf die jeweilige Organisationseinheit sollte es eine verantwortliche Person geben, bei der alle Kompetenzen bzgl. FDM zusammenlaufen und die Anreize im Prozess schaffen oder verstärken kann. Dies umfasst einerseits die Verantwortung über die Datenmanagementsysteme, andererseits die Verantwortung über die Daten selbst. Ebenso zählen die Ausarbeitung von Datenmanagementplänen mit den Anwenderinnen und Anwendern und das periodische Überprüfen des Dateninventars nach dunklen Daten zu den Aufgaben der verantwortlichen Person. Die verantwortliche Person sollte Kompetenzen bündeln, als Multiplikatorin bzw. Multiplikator fungieren und Wissen in Form von Schulungen weitergeben. Ebenso kann Verantwortung für dunkle Daten übernommen werden, die von de-registrierten Nutzerinnen und Nutzern stammen. Die konkrete Rolle des spezifischen Datenkurators wird in Kapitel 5.4 weiter zur Position des *Scientific Data Officer* ausformuliert.

4.2.5 Auswahlkriterien

Ressourcenökonomisch ist es in den computergestützten Simulationswissenschaften weder realistisch noch sinnvoll, *alle* erzeugten Daten aufzubewahren und zu speichern. Daher müssen Entscheidungs- bzw. Auswahlkriterien eine Anforderung sein. Diese geben Aufschluss über die Haltefristen und den Wert der Daten – also wie lange welche Dateien gespeichert werden sollten. Dabei muss zwischen Datenmenge und Wichtigkeit der Daten abgewogen werden. Hierbei muss insbesondere auch auf dunkle Daten, die durch De-Registrierung und Inaktivität entstehen, eingegangen werden. Die konkreten Auswahlkriterien und auch Haltefristen der (dunklen) Daten werden in Kapitel 5.5 expliziert.

4.3 Zusammenfassung und Diskussion der Anforderungen

Gemäß dem typischen Vorgehen des *Requirement Engineerings*, wie es in Kapitel 1.6 definiert wurde, erfolgt nun die Bündelung der Anforderungen zu einer Spezifikation in Form einer Anforderungsliste bzw. als Lastenheft. Nicht alle, oben natürlichsprachig aufgestellten, Anforderungen lassen sich als atomische Einheit formulieren. Die hier im Folgenden aufgestellte Anforderungsliste spiegelt dabei die Anforderungen wieder, die im Rahmen der Problemexplikation als kritisch für ein gelingendes Forschungsdatenmanagement ermittelt wurden. Diese Anforderungsliste ist Referenzrahmen für die folgenden Kapitel und insbesondere für die Evaluation in Kapitel 6.

4.3.1 Anforderungsliste

Funktionale Anforderungen an das Forschungsdatenmanagement

FA 1 Metadaten

- FA 1.1 Metadaten müssen Suchkriterien (sowohl fachspezifisch als auch allgemein) zur Verfügung stellen.
- FA 1.2 Metadaten müssen relevante und präzise Attribute enthalten.
- FA 1.3 Metadaten müssen nachvollziehbare Forschung durch Prozessinformation garantieren.
- FA 1.4 Metadaten müssen fachspezifische Informationen enthalten.
- FA 1.5 Metadaten müssen deskriptive Informationen enthalten.
- FA 1.6 Metadaten müssen technische Informationen enthalten.
- FA 1.7 Metadaten müssen automatisch extrahiert werden, sofern möglich.

FA 2 Integration in den Arbeitsprozess und Nutzerschnittstelle

- FA 2.1 Die Nutzerschnittstelle muss es ermöglichen, Daten einzupflegen, zu suchen, zu verschieben und zu browsen.
- FA 2.2 Die Nutzerschnittstelle muss Metadatenannotation ermöglichen.
- FA 2.3 Die Schnittstelle muss als Kommandozeilentool verfügbar sein.
- FA 2.4 Die Schnittstelle muss als grafische Oberfläche verfügbar sein.
- FA 2.5 Die Metadatenerfassung muss automatisiert nach der Simulation ausführbar sein.
- FA 2.6 Das Einspielen der Daten und Metadaten in ein Repositorium muss automatisiert nach der Simulation ausführbar sein.

FA 3 Skalierbarkeit muss sichergestellt werden, d.h. das System muss mit Datenobjekten im Größenbereich von Terabytes oder Petabytes umgehen können.

FA 4 Datensicherheit und -integrität

- FA 4.1 Die Daten müssen gemäß einer definierten Frist gespeichert werden.
- FA 4.2 Die Daten müssen auf mindestens zwei Kopien gespeichert werden.
- FA 4.3 Die Unveränderlichkeit der Daten muss überprüfbar sein.
- FA 4.4 Ende-zu-Ende-Datenintegrität muss gewährleistet sein.

FA 5 Organisatorische Anforderungen

- FA 5.1 Die Nutzerschnittstelle muss einen intrinsischen Anreiz liefern.
- FA 5.1 Datenmanagementpläne müssen zur Voraussetzung erhoben werden.
- FA 5.2 Die beteiligten Interessengruppen müssen in FDM geschult werden.

FA 5.3 Eine neue Rolle, welche die Verantwortung auf Ebene einer Organisationseinheit für alle Belange im FDM trägt, muss eingeführt werden.

FA 5.4 Diese Position muss extrinsische Anreize verstärken .

FA 5.5 Auswahlkriterien, wann und welche Dateien gelöscht werden können, müssen definiert und vom System berücksichtigt werden.

Nichtfunktionale Anforderungen

NFA 1 Metadaten

NFA 1.1 Automatisierte Metadatenextraktion muss ohne Administratorprivilegien lauffähig sein.

NFA 1.2 Automatisierte Metadatenextraktion muss so wenig Zusatzsoftware wie möglich benötigen.

NFA 1.3 Metadatenextraktion muss lokal in einem HPC-Workspace stattfinden können.

NFA 1.4 Die Ausführungszeit der automatisierten Metadatenextraktion muss im Verhältnis zur Ausführungszeit der Simulation stehen (typischerweise wenige Sekunden bis Minuten für die Metadatenextraktion).

NFA 2 Nutzerschnittstelle

NFA 2.1 Ortstransparenz muss möglich sein, um vom physikalischen Speicherort der Daten zu abstrahieren.

NFA 2.2 Die Nutzerschnittstelle muss sich einfach in den HPC-Workflow integrieren lassen.

4.3.2 Diskussion der Anforderungen

Im weiteren Verlauf der Dissertation wird der Fokus auf spezifische Anforderungen gelegt. Den Kern der Dissertation bilden die Arbeiten an den Anforderungen bzgl. Metadaten (FA 1 und NFA 1) in Kapitel 5.2, der Integration in die Arbeitsprozesse bzw. die Nutzerschnittstelle (FA 2, FA 5.1 und NFA 2) in Kapitel 5.3 sowie die Arbeit an einzelnen organisatorischen Anforderungen, wobei hier insbesondere FA 5.3 (in Kapitel 5.4) sowie FA 5.5 (in Kapitel 5.5) im Zentrum stehen. Die Anforderungen bzgl. Skalierbarkeit (FA 3) bilden den Projektrahmen und das Systemmodell der Dissertation und sind in Kapitel 5.1 dargestellt. Die Anforderungen, die sich auf Datensicherheit und -integrität beziehen (FA 4), werden nicht behandelt, da sich diese Anforderungen mit der gegenwärtig genutzten Speichertechnologie realisieren lassen und somit keine wissenschaftliche Relevanz aufweisen, allerdings der Vollständigkeit halber hier aufgeführt werden.

Kapitel 5

Konzepte für Forschungsdatenmanagement im HPC-Umfeld

Das vorliegende Kapitel zeichnet den Beitrag des Autors zum Forschungsdatenmanagement in den computergestützten Ingenieurwissenschaften nach und ist der zentrale Teil der Dissertation. Die in Kapitel 4 an Forschungsdatenmanagement formulierten Anforderungen wurden im Rahmen der Dissertation in diesem Kapitel in einen Konzeptentwurf und somit in Strategien gegen dunkle Daten überführt und schließlich als Lösungen implementiert. Wurde in Kapitel 4 also die Frage erörtert, *was* gelöst werden muss, wird in diesem Kapitel vorgestellt, *wie* es gelöst wird, um Forschungsdatenmanagement zur Verhinderung von dunklen Daten auszugestalten. Dies entspricht somit dem Teil des Systementwurfs gemäß dem Pflichtenheft. Dazu wird auf der technischen Seite in Unterkapitel 5.1 das Systemmodell besprochen, in dem alle weiteren technischen Konzepte eingebettet sind. Im folgenden Unterkapitel 5.2 wird die Entstehung von *EngMeta*, des Metadatenmodells für die Ingenieurwissenschaften, hergeleitet und alle Aspekte des Modells werden beleuchtet. Daran anschließend wird in 5.3 auf die automatisierte Metadatenerfassung und damit auch auf die Integration in den HPC-Workflow eingegangen. Auf der organisatorischen Seite wird die Rolle des Scientific Data Officers als spezifischen Datenkurator im HPC in Unterkapitel 5.4 hergeleitet und diskutiert. Schließlich werden in Unterkapitel 5.5 Entscheidungskriterien vorgestellt, um Haltefristen für Daten klar bestimmen zu können.

5.1 Technisch-organisatorischer Rahmen und Systemmodell

Die Arbeiten am Metadatenmodell und an der Metadatenerfassung fanden im Rahmen des Projekts DIPL-ING (Datenmanagement in Infrastrukturen, Prozessen und Lebenszyklen in den INGenieurwissenschaften) statt. DIPL-ING war ein Kooperationsprojekt zwischen dem HLRS, der Universitätsbibliothek Stuttgart (UB), den Technischen Informations- und Kommunikationsdiensten (TIK), dem Institut für

Aerodynamik und Gasdynamik (IAG) sowie dem Institut für Technische Thermodynamik und Thermische Verfahrenstechnik (ITT). Es wurde vom Bundesministerium für Bildung und Forschung unter dem Förderkennzeichen 16FDM008 gefördert und lief vom 1.4.2017 - bis zum 30.6.2019. Ziel des Projekts waren die Erarbeitung von Konzepten zum Forschungsdatenmanagement in den computergestützten Ingenieurwissenschaften. Die Anforderungsanalyse für die Fachbereiche Thermodynamik und Aerodynamik, die in (Iglezakis und Schembera, 2018) veröffentlicht wurde, war der Ausgangspunkt für die weitere Projektarbeit. Neben der Entwicklung des Metadatenmodells *EngMeta*, welches auch Kernthema der vorliegenden Dissertation ist, wurde im Projekt das Kompetenzzentrum FoKUS[81] für die Universität Stuttgart eingerichtet. Dessen Aufgabe ist die Schaffung und Verstetigung von Ansprechpartnerinnen und Ansprechpartnern sowie die Bündelung der Kompetenzen für alle Fragen, die das Forschungsdatenmanagement auf Universitätsebene betreffen. Darüber hinaus wurden im Rahmen des Projekts ein Forschungsdatenrepositorium auf Basis von Dataverse aufgebaut[82]. Da neben dem technischen Aufbau von FDM auch die organisatorischen Angriffspunkte thematisiert wurden, wurde innerhalb des Projekts eine eintägige Schulung für Doktorandinnen und Doktoranden am 20.3.2019 durchgeführt, an der auch der Autor der Dissertation beteiligt war.

Die Systemarchitektur, die für DIPL-ING verwendet wird und in die die Arbeiten zum Metadatenschema *EngMeta* und zur Metadatenerfassung eingebettet sind, wird im vorliegenden Unterkapitel vorgestellt. Sie dient dabei als Referenz und konzeptionelle Orientierung, um Forschungsdatenmanagement den Anforderungen gemäß auszugestalten. Dabei wird an der Unterteilung in drei Schichten, wie sie einem allgemeinen Softwarentwurfsmuster entspricht, festgehalten. Diese Drei-Schichten-Architektur liegt vielen Forschungsdatenmanagementsystemen zugrunde[83]. Dabei wird von den folgenden drei Schichten ausgegangen: Die Speicherschicht ist für die physikalische Speicherung der Daten zuständig und ist die Basiskomponente, auf der alles Weitere aufbaut. Die Objektschicht übernimmt Datenverwaltungsaufgaben und macht die Daten zu Objekten, die such- und findbar sind. Die Nutzungsschicht ist die oberste Schicht und vermittelt zwischen dem System und den Nutzerinnen und Nutzern.

Die Drei-Schichten-Architektur ermöglicht Skalierbarkeit und Flexibilität durch die Trennung von Speicherort, Verwaltung und Interface. Indem die Datenverwaltung z.B. in Form einer Datenbank mit indizierten Metadaten im Objekt-Layer implementiert ist, müssen beim Suchen nach Daten nicht die Daten jeweils selbst eingelesen, sondern nur die Datenbankeinträge gesucht werden. Erst bei erfolgreicher Suche können die Daten dann aus der Speicherschicht gelesen und zur Weiterverarbeitung bereitgestellt werden. Die Objektschicht ist abstrakt, also ein Verzeichnis der Daten,

[81] `https://www.izus.uni-stuttgart.de/fokus/`, Zugriff 7.1.2019.
[82] `https://darus.uni-stuttgart.de/`, Zugriff 7.1.2019.
[83] Siehe Kapitel 2.3.3.

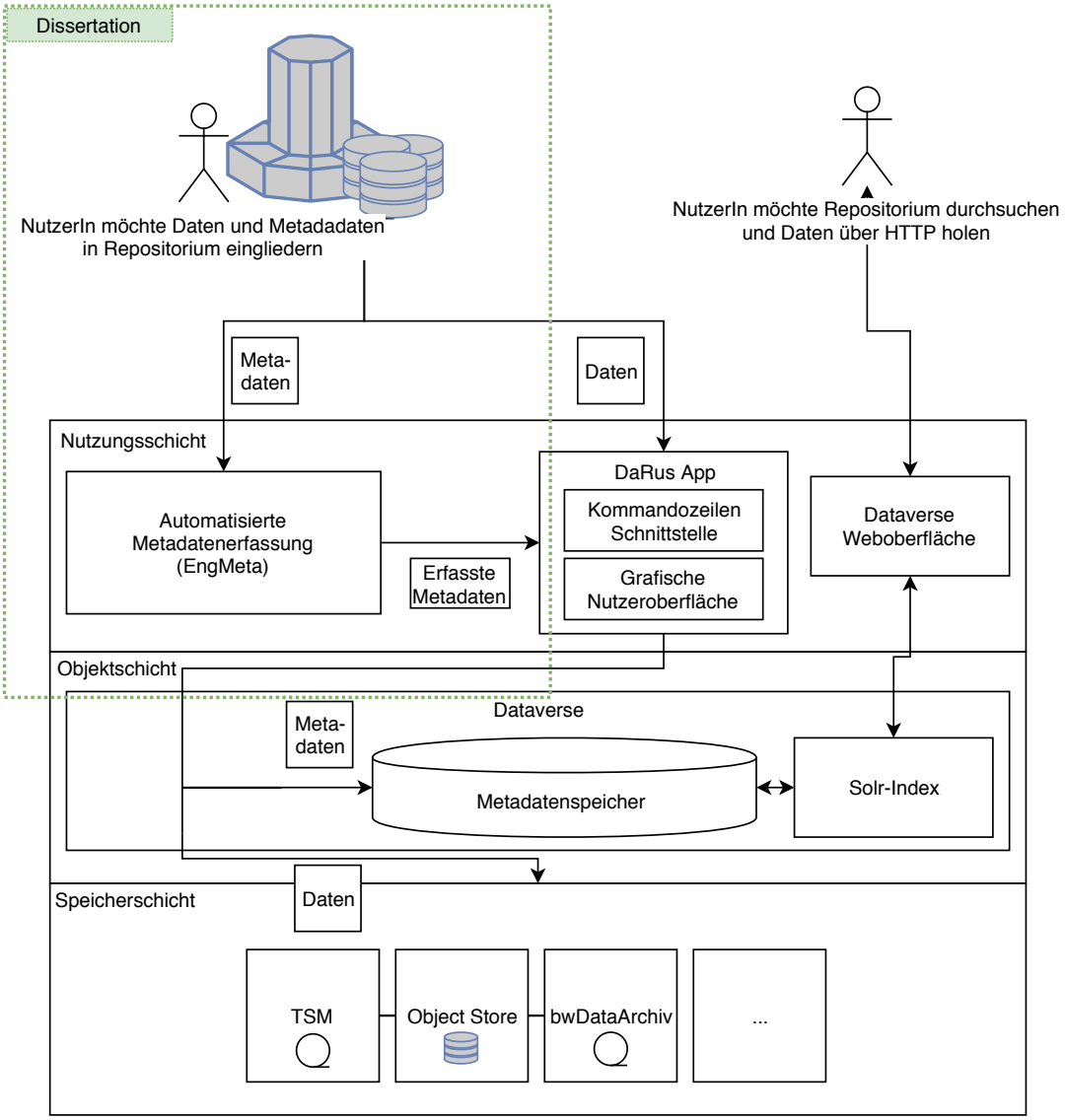

Abbildung 5.1: Die Drei-Schichten-Architektur mit Speicherschicht, Objektschicht und Nutzerschicht. Der Beitrag des Autors ist grüngestrichelt umrandet.

welches Referenzen zu den jeweiligen Speicherorten der Daten in der Speicherschicht hält. Durch die Trennung in die Speicher- und Objektschicht mit deren jeweiligen Aufgaben der Datenspeicherung und Datenverwaltung trägt die Architektur maßgeblich zur Erfüllung der Anforderung *FA3 Skalierbarkeit* bei. Die Entkopplung von Speicherort und Verwaltung findet sich in den meisten Datenmanagementsystemen aus dem HPC-Umfeld wieder, da nur so die großen Datenmengen entsprechend gespeichert werden können. Dies wird bereits in Kapitel 2.5 dargestellt und dient auch als Basis für alle weiteren Überlegungen. Im Folgenden werden die drei Schichten für den Kontext der Dissertation genauer beschrieben und Implementierungsvorschläge gegeben. Die Architektur und der Zusammenhang zwischen den Schichten ist in Abbildung 5.1 dargestellt. Dabei sind die jeweiligen konkreten in der Dissertation genutzten Komponenten eingezeichnet sowie die Datenflüsse, die für die Dissertation relevant sind. In

dieser Abbildung ist der Beitrag, der innerhalb des Dissertationsvorhabens entstand, bezogen auf die Systemarchitektur und den Datenfluss grüngestrichelt hinterlegt. Der Beitrag des Autors ist hier vor allem die Integration in die HPC-Umgebung, die Implementierung und Integration der automatisierten Metadatenerfassung sowie die Interaktion mit dem Repositorium. Die Objektschicht wurde nur insofern mitentwickelt, wie Anforderungen aus dem HPC und aus der Metadatenerfassung heraus formuliert wurden.

Speicherschicht Die Speicherschicht ist die logische Basiskomponente für alle weiteren Teile des Forschungsdatenmanagementsystems, wobei die physikalische Speicherung der Daten auf dieser Ebene stattfindet. Um Skalierbarkeit im Speicherplatz zu erreichen, muss sie aus einfach erweiterbarem Speicher aufgebaut sein. Dabei ist Bandspeicher die einzige Speicherform, die die Anforderung der großen Datenmengen erfüllt (FA 3) und die geringsten Gesamtbetriebskosten besitzt (Faulhaber, 2015). Auf dieser Schicht wird auch die Bitstream Preservation, d.h. die Unveränderlichkeit der hinterlegten Daten, sichergestellt. Dies wird über zwei oder mehrere Kopien an verschiedenen Standorten erreicht oder durch die Nutzung von Checksummen. Darüber hinaus muss die Speicherschicht verteilbar sein, d.h. sich über mehrere physikalisch verteile Standorte und Technologien erstrecken können. Es können verschiedene Speichertechnologien eingesetzt werden, wie z.B. Bandspeicher, Festplattenspeicher oder SSDs. Die Art der Medien richtet sich dabei nach dem Zweck der Nutzung. Bandspeichermedien kommen überall dort zum Zug, wo die Daten kalt, d.h. nur selten in Benutzung sind, wohingegen Festplattenspeicher für Daten genutzt werden sollte, auf die häufig zugegriffen wird.

In der vorliegenden Implementierung findet sich ausschließlich das Objektspeichersystem des TIK angebunden. Dieses besteht aus einem NetApp StorageGRID mit S3-Schnittstelle, das momentan Festplattenspeicher über zwei Standorte verteilt benutzt. Es ist in der Lage, auch Bandspeicher oder weitere universitätsinterne Speichermöglichkeiten, wie z.B. TSM, anzubinden.

Objektschicht Die Objektschicht ist die zentrale Komponente eines Forschungsdatenmanagementsystems und entspricht im Wesentlichen den Funktionen einer Datenbank. Im vorliegenden Entwurf wird die Funktion von der Repositoriumssoftware Dataverse übernommen, die im Rahmen des DIPL-ING- Projektes prototypisch aufgesetzt wurde und Mitte 2019 in den Produktivbetrieb überführt werden soll[84]. Die Metadatenverwaltung in Dataverse basiert auf der relationalen Datenbank PostgreSQL, wobei die Suchfunktion mittels *Solr* abgewickelt wird. Dataverse läuft innerhalb eines *Jave EE Application Servers*. Die Objektschicht als zentrale Komponente des Repositoriums bietet die Möglichkeit, die Daten privat zu halten oder zu veröffentlichen, womit eine DOI-Vergabe einhergeht.

[84] `https://www.izus.uni-stuttgart.de/fokus/darus/`, Zugriff 23.3.2019.

Nutzungsschicht Die Nutzungsschicht ist die Präsentationsschicht des Systems und stellt die Schnittstelle zur Verfügung, über die die Nutzerinnen und Nutzer auf das System zugreifen können. Dies umfasst das Suchen, Browsen, Einspielen, Zurückspielen und Verändern von Daten und Metadaten. Dabei muss die Nutzungsschicht besonders gut in den Arbeitsprozess integrierbar sein, um die Hemmschwelle der Nutzung tief zu setzen.

Dazu wurde im Projekt DIPL-ING die *DaRUS-App* entwickelt, welche ein Kommandozeilen-Tool und eine grafische Benutzeroberfläche zum Einspielen von Daten und Metadaten zur Verfügung stellt. Diese spricht die API von Dataverse[85] in der Objektschicht an. Die DaRUS-App stellt somit die Schnittstelle ins DaRUS-Repositorium in Form eines Programms dar und ist eine Java-Anwendung, die HTTP-Anfragen an das Repositorium umsetzt. Dataverse verfügt auch über eine Webschnittstelle[86], in der nach Daten gesucht werden kann und in der sich auch Daten hoch- und herunterladen lassen.

Ebenfalls ist die automatisierte Metadatenerfassung, auf die im Unterkapitel 5.3 genauer eingegangen wird, in dieser Schicht lokalisiert. Die Ergebnisse der automatisierten Metadatenerfassung können dann mittels der DaRUS-App mit den Daten zusammen in das Dataverse-Repositorium als Datensatz eingespielt werden. In der Evaluation wird in Kapitel 6.4.2 gezeigt, wie sich der Ablauf *Metadatenerfassung – Einspielen der Daten+Metadaten* beispielsweise mittels Epilog-Skripten vollständig automatisieren lässt.

5.2 *EngMeta* – Ein Metadatenmodell für die Ingenieurwissenschaften

Wie sich in den Kapiteln 1.3.1 und 3 bereits zeigte, ist das Fehlen einer passenden Beschreibung von Forschungsdaten die größte Herausforderung für das Forschungsdatenmanagement und der maßgebende Faktor für das Entstehen dunkler Daten. Daher wurden Metadaten als Hauptkonzept für gelingendes Forschungsdatenmanagement zur Anforderung erhoben[87]. Gemäß (Edwards u. a., 2011) darf ein Metadatenmodell nicht als fertiges Produkt verstanden werden, sondern als wissenschaftliche Kommunikationsform, die sich im Fluss befindet und später durch weitere Produkte wie die Metadatenerfassung und Rollen wie die des Scientific Data Officers ergänzt wird. Dabei ist es die Aufgabe des Metadatenmodells, den Objektbezug herzustellen, also die ingenieurwissenschaftlichen Forschungsdaten und ihre Eigenschaften einheitlich zu beschreiben. Dazu wurde der gesamte Forschungsprozess und die darin produzierten Daten analysiert. Des Weiteren wurde festgelegt, was für eine umfassende Beschreibung von Forschungsdaten relevant ist. Jegliche weiteren Arbeiten, wie z.B. die des prototypischen Aufbaus des Repositoriums oder die Konzeption der

[85] `http://guides.dataverse.org/en/latest/api/index.html`, Zugriff 23.3.2019.
[86] `https://darus.uni-stuttgart.de/`, Zugriff 10.2.2019.
[87] Siehe Anforderungsblock FA 1 in Kapitel 4.3.

automatisierten Metadatenerfassung, basieren schließlich auf den Konventionen, die durch das Metadatenmodell festgelegt werden.

Die Entwicklung des Metadatenmodells erfolgte dabei innerhalb des Projekts DIPL-ING zusammen mit Projektpartnerinnen und Projektpartnern aus den Ingenieurwissenschaften sowie aus der Universitätsbibliothek. Die Kooperation ist in den Kapiteln 1.5 und 5.1 aufgeschlüsselt und stellte sich bezogen auf das Metadatenmodell wie folgt dar: Der wissenschaftliche Prozess wurde qualitativ mit den Vertreterinnen und Vertretern der Ingenieurwissenschaften analysiert und in Anforderungen an das Metadatenmodell überführt. Neben dieser fachwissenschaftlichen Perspektive kamen die bibliografischen Anforderungen nach Standardisierung aus der Universitätsbibliothek, um die Datenbeschreibung in Teilen kompatibel und interoperabel zu Repositorien zu halten oder um die DOI-Vergabe zu ermöglichen. Diese bilden die bibliothekswissenschaftliche Perspektive. Vom Autor stammt die aus der Informatik kommende Perspektive, Metadaten als eine Ontologie zu modellieren, die eine umfassende Beschreibung des Simulationsprozesses durch Relationen zwischen Objekten darstellt. Dort spielen, neben den fachspezifischen, die Prozessmetadaten (also die Rechnereigenschaften und Informationen, wie die Simulation ablief) eine entscheidende Rolle bei der Datendokumentation.

Im Folgenden soll die Konzeption und Entstehung des Modells nachgezeichnet werden. Dazu wurde von einem Objektmodell ausgehend, welches das Universum beschreibt, in dem ingenieurwissenschaftliche Forschungsdaten erzeugt werden, das datenobjektzentrische Metadatenmodell *EngMeta* (Engineering Metadata) entworfen. Die Metadaten richten sich dabei nach den in Kapitel 4.1.1 definierten vier Kategorien und sind im Folgenden mit Beispielen für den vorliegenden Anwendungsfall der computergestützten Ingenieurwissenschaften dargestellt:

- **Deskriptive Metadaten** beschreiben die Forschungsdaten allgemein, z.B. über *Schlagwörter*, einen *Titel* oder Informationen über die *Erstellerinnen und Ersteller*.
  Im Folgenden farblich dargestellt in Ocker.

- **Technische Metadaten** beschreiben technische Eigenschaften der Forschungsdaten wie z.B. den *Dateinamen*, die *Dateigröße*, *Checksumme* oder *Speicherort*.
  Im Folgenden farblich dargestellt in Blau.

- **Prozessmetadaten** beschreiben den Entstehungsprozess der Forschungsdaten. Zu den Prozessmetadaten gehören z.B. Informationen über die *Rechnerumgebung* oder die verwendete *Software*.
  Im Folgenden farblich dargestellt in Dunkelgrün.

- **Fachspezifische Metadaten** beschreiben Forschungsdaten aus der Perspektive der Wissenschaftsdomäne detailliert fachlich. Dies umfasst z.B. Informationen über das *Zielsystem* oder die *zeitliche Auflösung* der Simulation.
  Im Folgenden farblich dargestellt in Grasgrün.

Das Objektmodell, welches sich nach diesen Kategorien richtet, wurde schließlich in einem XML-Schema implementiert. Die Erläuterungen zur Vorgehensweise der Schritte *Explikation der relevanten Objekte – Explikation ihrer Objekteigenschaften – datenzentrisches Objektmodell* (Abschnitt 5.2.1) – *Metadatenmodell* (Abschnitt 5.2.2) – *XML-Schema* (Abschnitt 5.2.3) finden sich entlang des Beispiels des relevanten Objekts *Rechnerumgebung* nachgezeichnet.

5.2.1 Objektmodell

Metadaten stellen den Objektbezug her, wobei zentral festgestellt werden muss, welches Objekt oder welche Teile davon beschrieben werden sollen. Die Explikation der Eigenschaften, die durch Metadaten repräsentiert werden sollen, ist hierbei die erste Aufgabe. Die Objekte, die im Fall der vorliegenden Dissertation modelliert werden, sind Forschungsdaten aus den computergestützten Ingenieurwissenschaften. Um zu einer umfassenden Beschreibung von Forschungsdaten zum Zwecke der Datendokumentation zu kommen, müssen zunächst alle für die Beschreibung relevanten Charakteristika des Forschungsprozesses gewählt und in einem Objektmodell dargestellt werden. Da allein die Beschreibung von einzelnen Charakteristika der Daten selbst, wie z.B. reine Dateieigenschaften, nicht ausreicht, müssen auch Informationen enthalten sein, die das simulierte Zielsystem beschreiben und ebenso Informationen darüber, wie sie entstanden sind. Dieser Zusammenhang ist in Abbildung 5.2 dargestellt. Die für die Beschreibung relevanten Teile des Forschungsprozesses sind neben den Daten selbst und ihren Eigenschaften, welche die technischen Metadaten liefern, auch Informationen über die Modellimplementierung und die Simulation. Dies beinhaltet beispielsweise Informationen über die genutzte Hard- und Software, d.h. die Prozessinformationen. Darüber hinaus müssen auch Informationen über das simulierte Zielsystem und das Modell festgehalten werden, was die fachspezifischen Metadaten konstituiert. Nach der Analyse und Explikation der relevanten Teile des Forschungsprozesses folgt die Überführung in ein Objektmodell. Dabei repräsentiert eine *Entität* (bzw. ein *Objekt*) im Objektmodell eine abgeschlossene logische Teileinheit, die durch gewisse Eigenschaften, genannt *Attribute*, beschrieben wird oder sich ihrerseits wieder durch Unterobjekte zusammensetzt. Eine Entität bezieht sich dabei auf Teile des Forschungsprozesses, wobei die Entitäten in der Regel zueinander in Beziehung stehen.

Explikation der Objekte

Als erster Schritt muss geklärt werden, welche Entitäten bzw. Objekte für die Beschreibung der Daten relevant sind. Die Identifikation aller, für Forschungsdatenmanagement in den Simulationswissenschaften relevanten, Entitäten wurde in enger Zusammenarbeit mit den beteiligten Akteuren der jeweiligen Anwendungsfälle durchgeführt. Ein Beschreibungsmodell für Forschungsdaten kann sich nicht auf

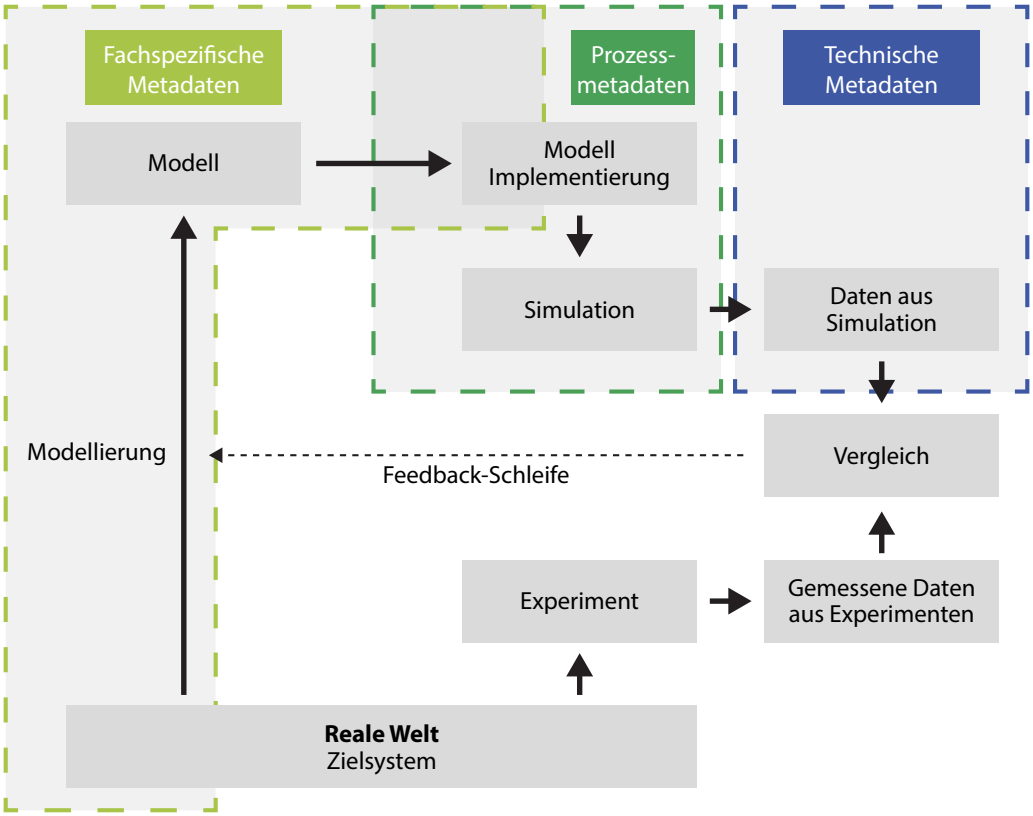

Abbildung 5.2: Metadaten müssen mehrere Schritte des Simulationsprozesses beschreiben. Dargestellt ist die Zuordnung der Schritte zur jeweiligen Art der Metadaten. Deskriptive Metadaten sind nicht eingezeichnet, da sie prinzipiell alle Bereiche betreffen, diese Beschreibung aber von höherer Ebene aus geschieht.

einzelne, für sich selbst stehende Entitäten beschränken. Damit wären nur die Eigenschaften eines einzelnen Objektes erfasst, aber beispielsweise nicht die Eigenschaften über dessen Entstehungsprozess. Dies bedeutet insbesondere, dass weitere Merkmale hinzugenommen werden mussten, die sich auf das Objekt beziehen, aber außerhalb der Eigenschaften des Objekts selbst liegen. Die einzelnen Objekte müssen sich aufeinander beziehen und bilden so eine umfassende Beschreibung aller, für das Forschungsdatenmanagement in den Ingenieurwissenschaften relevanten, Arbeitsschritte, entstandenen Daten und modellierten Systeme. Das Ergebnis des ersten Schrittes findet sich in Abbildung 5.3, die das Verhältnis der Objekte im Forschungsprozess zueinander zeigt sowie ihre Zuordnung zu den jeweiligen Metadatenkategorien. Hierbei ist dargestellt, dass die Simulation, die erzeugten Daten und das simulierte Zielsystem eine zentrale Rolle spielen. Die Simulation erzeugt die Daten, die das simulierte Zielsystem repräsentieren. Sie wird dabei von einer Person ausgeführt und nutzt HPC-Systeme, Software und eine spezifische Methodik.

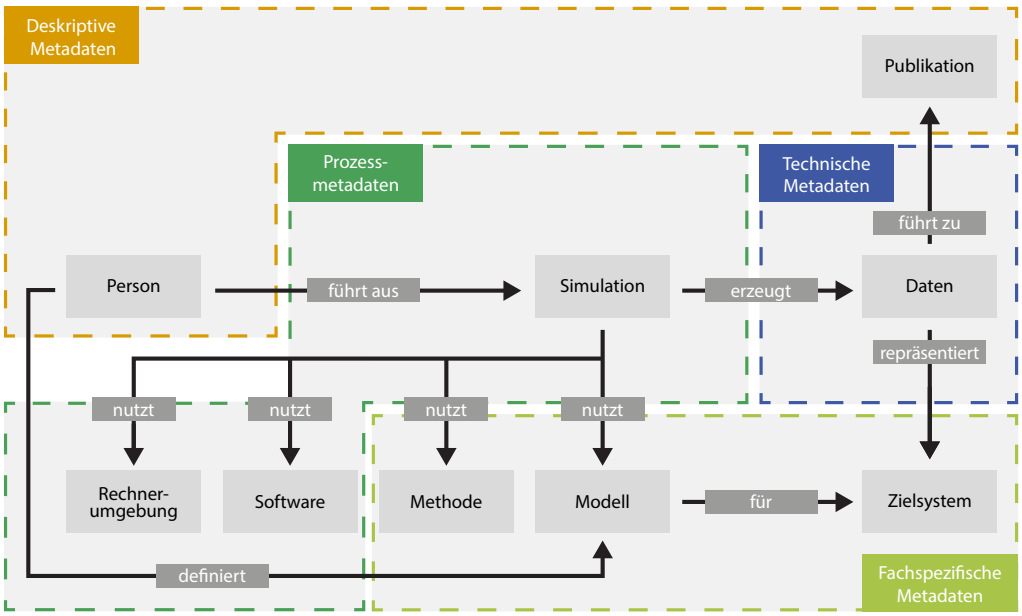

Abbildung 5.3: Die zentralen Elemente *Simulation*, *Daten* und *Zielsystem* sowie weitere explizierte Objekte im Forschungsprozess und deren Zuordnung zu den vier Beschreibungskategorien sind farblich codiert.

Explikation der Objekteigenschaften

Nachdem nun die Objekte expliziert wurden, sind die Relationen zwischen den Objekten und die zentralen Objekte *Simulation*, *Daten* und *Zielsystem* klar. Diese Darstellung muss nun in einem zweiten Schritt weiter gruppiert und die Objekteigenschaften müssen herausgearbeitet werden.

Der Teil *Daten* im Forschungsprozess (siehe Abbildung 5.3) besteht in der Realität beispielsweise aus einer oder mehreren Dateien, weswegen eine Entität *Datei* definiert werden muss, die wiederum durch ihre Größe, ihr Erstellungsdatum usw. charakterisiert, also über die Attribute *Größe* und *Erstellungsdatum* definiert ist. Diese Teile der Beschreibung werden den technischen Metadaten zugeordnet. Ebenso ist für die *Simulation* (bzw. ihre Ausführung) das verwendete HPC-System bzw. die Rechnerumgebung von Bedeutung. Somit muss ein Beschreibungsmodell für Forschungsdaten die Entität *Rechnerumgebung* beinhalten, welche sich durch Attribute wie *Cores*, *CPU*, *Compiler* usw. definiert. Dabei kann der *Compiler* noch einmal als eigene Entität aufgefasst werden, die durch Attribute wie *Version*, *Name*, *Flags* charakterisiert wird. Diese Verschachtelung der Objekte ist in Abbildung 5.4 beispielhaft für die Entität *Rechnerumgebung* dargestellt. Diese Entitäten und Attribute werden den Prozessmetadaten zugeordnet.

Von hoher Priorität ist die Modellierung von fachspezifischen Merkmalen, die sich vor allem in der Beschreibung des beobachteten *Zielsystems*, der *Methodik* und der Beschreibung des *Modells* ausdrücken, wie in Abbildung 5.3 als fachspezifische Metadaten dargestellt. Sowohl in der Thermodynamik als auch in der Aerodynamik

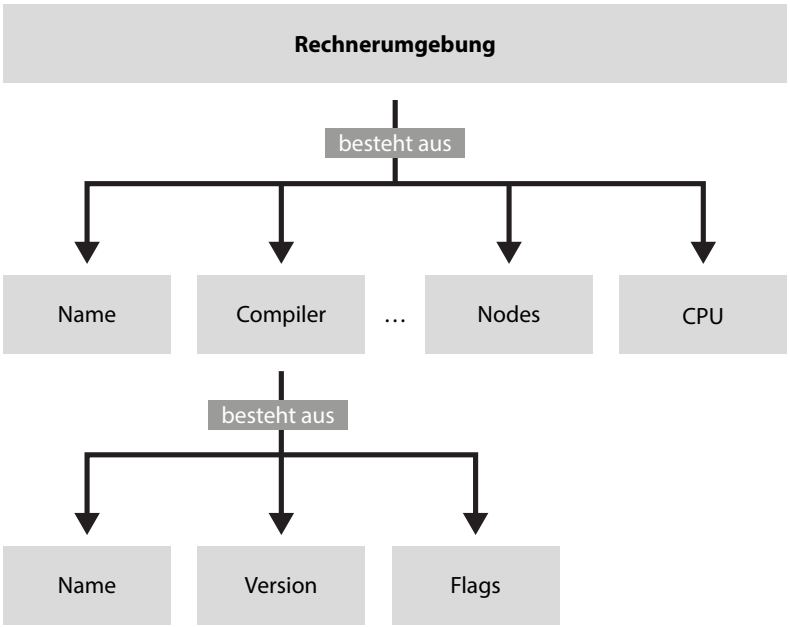

Abbildung 5.4: Darstellung der Entität *Rechnerumgebung* mit ihren Unterobjekten.

wird das zu analysierende Zielsystem mittels gewisser Parameter und Variablen modelliert, wobei dies *kontrollierte Variablen* und *gemessene Variablen* sein können. Die kontrollierten Variablen sind diejenigen, die von Seite der Wissenschaftlerin oder des Wissenschaftlers definiert werden, aber nicht Gegenstand der eigentlichen Messung sind. Im Bereich der Thermodynamik können dies beispielsweise Druck, Temperatur, Thermostat oder Barostat sein. Die gemessenen Variablen sind diejenigen Variablen, die von Interesse sind und die eigentlichen Ergebnisse der Simulation darstellen. Sie entsprechen den Messungen bei Experimenten. Als *Parameter* legen die Variablen Einflussgrößen fest und parametrisieren die Simulationsmethode. Sie sind bedeutend zur Rekonstruktion und Nachvollziehbarkeit von Simulationen. Diese drei Variablenformen sind essenziell für die Simulation im Bereich der Ingenieurwissenschaften und mussten somit in einem Beschreibungsmodell Platz finden. Ebenso ist die *zeitliche Auflösung* und die *räumliche Auflösung*, in der sich eine Simulation abspielt, wesentlich für die Beschreibung ihrer Ausgabedaten. Die zeitliche Auflösung trifft dabei den Kern der Definition von Simulation, nämlich die Imitation eines Systems oder Prozesses über einen gewissen Zeitraum. Die räumliche Dimension ist ebenso wesentlich. Sie erfasst die räumliche Komponente der Simulation, welche durch die Physik vorgegeben ist. Die *Komponenten* sind spezifisch für die Thermodynamik und bezeichnen die Charakteristika der in der Simulation repräsentierten thermodynamischen Komponenten, wie z.B. der Kraftfelder. In ihrer Beschreibung sollte auch die *Simulationsmethode* erfasst werden, sie liefert übergeordnete Informationen, z.B. ob es sich um eine gleichungsbasierte oder agentenbasierte Simulation des Zielsystems handelt. In späteren Schritten kann dies auch die Analyse- oder Visualisierungsmethode abbilden.

Die Objektmodellierung ist dabei bewusst offen gewählt, um potentiell auch weiteren, außerhalb der Thermo- und Aerodynamik stehenden Fachgebieten eine Möglichkeit zur Verbesserung der Datendokumentation zu geben. Die zeitliche und räumliche Auflösung sowie Nutzung von kontrollierten und gemessenen Variablen ist den meisten Simulationen – egal aus welchem Fachgebiet innerhalb der Ingenieurwissenschaften – gemein. Ebenso ähneln sich die Nutzung einer bestimmten Rechnerumgebung und bestimmter Simulationsmethoden, weshalb die vorgenommene Objektmodellierung als im Fachbereich universell einsetzbar gelten kann. In Kapitel 6.2.2 wird darüber hinaus der Einsatz des Metadatenmodells im Hinblick auf die Anwendbarkeit in anderen Fachbereichen als den computergestützten Ingenieurwissenschaften evaluiert.

Datenzentriertheit des Objektmodells

In den Simulationswissenschaften nimmt der durch die Simulation erzeugte Datensatz, der einen Zustand oder einen Verlauf eines Zielsystems repräsentiert, die zentrale Stelle ein, weswegen auch das Modell entsprechend datenzentrisch aufgebaut ist. Dieser Gedanke findet sich beispielsweise auch im CERA-2.5-Metadatenmodell in den Klimawissenschaften wieder und ist den Charakteristika des simulationswissenschaftlichen Arbeitsprozesses geschuldet. Nach der Durchführung der Simulation wird der Datensatz nicht mehr weiter verändert, sofern der Simulationslauf erfolgreich war. Ist die Simulation beendet, kommen keine Daten mehr hinzu, d.h. die Daten selbst verändern sich nicht mehr. Nach erfolgreicher Simulation soll ein gewisser Stand festgehalten werden, der durch die Ausgabedaten der Simulation repräsentiert wird. Dies unterscheidet die Simulationswissenschaften z.B. von der Programmierung, im Rahmen derer es wichtig ist, die inkrementellen Änderungen am Code zu erfassen, da der Ausgangscode immer wieder direkt erweitert wird. Im Gegensatz dazu werden die Daten aus der Computersimulation selbst nicht mehr direkt verändert, können aber durchaus als Eingabedaten für Auswertungsalgorithmen, als Vorstufe zum eigentlichen Simulationslauf oder für neuerliche Simulationsläufe gelten[88]. Um allerdings auch diese einzelnen Schritte erfassen zu können, wurde die Entität *Verarbeitungsschritt* eingeführt, welche Informationen über die verwendeten Geräte, Rechner, Compiler usw. angibt, wobei ein Datensatz mehreren Prozessschritten zugeordnet sein kann. Damit sollen mit diesem Objektmodell auch die verschiedenen Stufen eines simulationswissenschaftlichen Arbeitsprozesses abgebildet (z.B. Schritt 1: Testlauf, Schritt 2: Produktionslauf, Schritt 3: Datenauswertung) und, sofern vorhanden, jeder dieser Schritte mit den jeweiligen Dateien assoziiert werden können.

[88] In der Molekulardynamiksimulation ist dies z.B. die Equilibrierung, die das System in seinen Gleichgewichtszustand bringt. Erst von diesem Zustand ausgehend sind Molekulardynamiksimulationen nur noch kleinen Schwankungen des Systems selbst ausgesetzt und können sinnvoll simuliert werden.

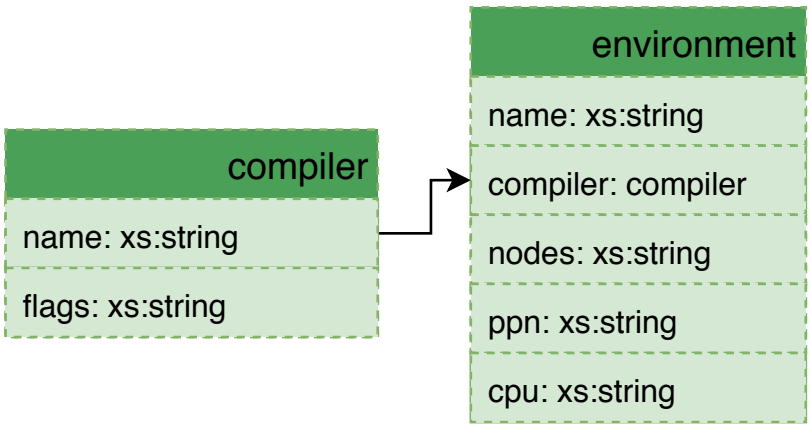

Abbildung 5.5: Darstellung der Entität *Rechnerumgebung* mit ihrem Unterobjekt *Compiler* und Wertebereichen nach XML-Konvention.

5.2.2 Vom Objektmodell zum Metadatenmodell

Die komplette Darstellung aller Objekte, ihrer Beziehungen und ihrer Attribute ist in Abbildung 5.6 zu finden. Dabei richtet sie sich nach den, in Kapitel 4 formulierten Anforderung, dass gut beschriebene Forschungsdaten deskriptive, technische, fachspezifische und Prozessmetadaten umfassen müssen. Diese Einordnung ist in Abbildung 5.6 farblich codiert dargestellt. Nach der abstrakten Modellierung als Objektmodell müssen nun in einem nächsten Schritt die Entitäten sowie die Relationen zwischen den Entitäten in Metadatenobjekte überführt werden. Beispielsweise wird die in Abbildung 5.4 dargestellte Objektrelation, die eine Rechnerumgebung beschreibt, wie in Abbildung 5.5 auf eine Metadatenrelation abgebildet und mit Wertebereichen versehen.

In diesem Schritt werden bestehende Metadatenmodelle auf ihre Anwendbarkeit für die vorliegenden Anwendungsfälle geprüft. Dabei stellt sich heraus, dass keines der bestehenden Metadatenmodelle alle als relevant bestimmten Entitäten umfasst. Die im Kapitel 2 als verwandte Arbeiten dargestellten Metadatenmodelle beschreiben jeweils nur einen Teilbereich. DataCite modelliert allgemein deskriptive Metadaten, um Daten zitierfähig zu machen, jedoch sieht es nur wenige technische Beschreibungsmöglichkeiten vor. So werden einige Teile in *EngMeta* DataCite-konform definiert. CodeMeta wird zur Beschreibung der Software genutzt und ExptML zur Beschreibung von Experimentalgeräten. PREMIS wird für einige technische Metadatenattribute genutzt. CERA-2.5 geht indirekt über das datenzentrische Design in *EngMeta* ein. Prov bzw. ProvONE wird nicht direkt genutzt, allerdings sind einige Metadatenfelder kompatibel mit diesem Standard[89]. Auch wird in diesem Schritt die, zunächst als aussichtsreich wirkende, Beschreibungssprache MSML zur Nutzung für unseren Anwendungsfall evaluiert. Dabei enthält die auf CML basierende

[89] Für diesen Standard wurde im Projekt DIPL-ING ein sog. *Crosswalk* entwickelt. Dieser definiert eine eindeutige Abbildung von einem Metadatenschema in ein anderes und ermöglicht so die Nutzung der Metadatenschlüssel in einem anderen Kontext, z.B. für Software oder Repositorien, die speziell Prov nutzen.

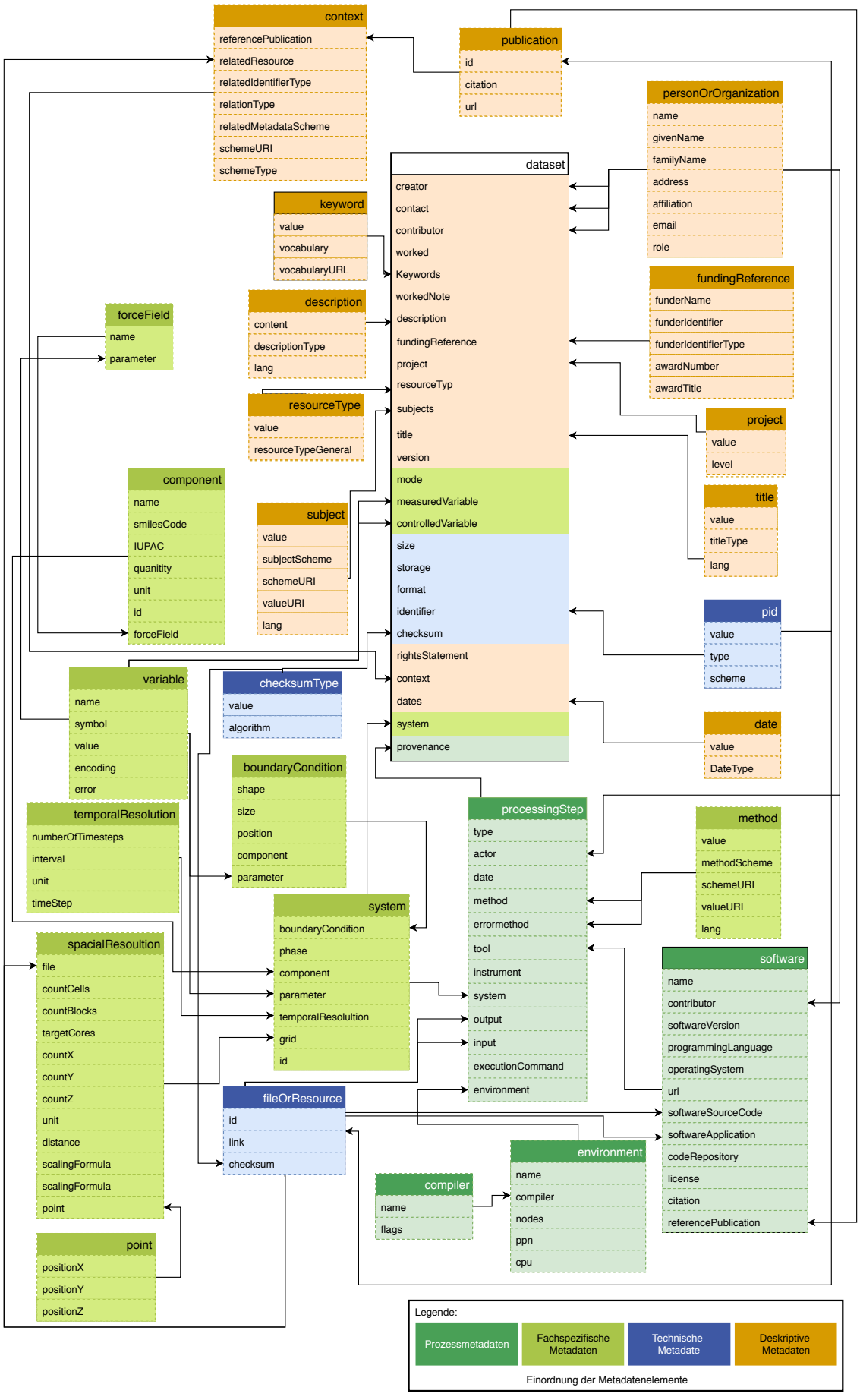

Abbildung 5.6: Das Objektmodell von EngMeta für die Beschreibung von Forschungsdaten. Darstellung aller Objekte und der Relationen zwischen den Objekten und Einordnung in die vier verwendeten Metadatenkategorien gemäß dem definierten Farbschema.

	deskriptiv	technisch	prozessual	fachspezifisch	Geht in EngMeta ein?
CERA-2.5	✓	(✓)	✓	✓	indirekt
CodeMeta	✓	(✓)	✓	–	ja
CML	✓	–	✓	✓	nein
DataCite	✓	✓	–	–	ja
Dublin Core	✓	✓	–	–	nein
ExptML	–	–	✓	✓	ja
PREMIS	✓	✓	✓	–	ja
Prov/ONE	✓	✓	✓	–	indirekt

Tabelle 5.1: Übereinstimmung der besprochenen Metadatenmodelle mit den jeweiligen Beschreibungskategorien bzgl. Forschungsdaten und Einfluss auf EngMeta.

MSML nur fachspezifische und Prozessmetadaten und modelliert den Workflow *vor* dem Simulationslauf. Das Ziel der MSML ist anders geartet als der Ansatz von EngMeta. Während EngMeta darauf abzielt, den Wissenschaftlerinnen und Wissenschaftlern eine Beschreibungsmöglichkeit zu bieten, die ihre fachbereichs-, instituts- oder gruppenspezifischen (oder komplett individuellen) Tools und Prozesse unverändert lässt und keinerlei Intervention vor bzw. in diesem Prozess erfordert, setzt MSML und MosGrid genau dort an. Die Wissenschaftlerinnen und Wissenschaftler sind angehalten, den Prozess schon vorab in Form der MSML zu beschreiben. Dieser Ansatz wird für das Vorhaben der Dissertation verworfen, da dies zusätzlichen Aufwand für die Wissenschaftlerinnen und Wissenschaftler bedeutet und die Hemmschwelle zur Metadatenannotation steigen würde. Diese beiden Faktoren gelten als wesentliche Herausforderungen für gelingendes Forschungsdatenmanagement, wie in Kapitel 1.3.2 diskutiert wurde. Der Einfluss der verschiedenen Metadatenmodelle auf *EngMeta* ist noch einmal in der Tabelle 5.1 zusammenfassend dargestellt. Die Überdeckung der Metadatenentitäten und -attribute aus *EngMeta* mit bestehenden Standards ist in Abbildung 5.7 dargestellt und farblich gekennzeichnet. Dabei steht jeweils das Präfix des entsprechend referenzierten Metadatenstandards den einzelnen Feldern voran[90].

Die fachspezifischen, d.h für die Thermo- und Aerodynamik charakteristischen, Metadatenentitäten sind in Tabelle 5.2 zusammengefasst. Dabei sind dies entweder einzelne Metadatenfelder oder ganze Metadatenblöcke, die aus mehreren Teilfeldern bestehen. Beispielsweise handelt es sich bei der Entität *environment*, im Gegensatz zu *worked*, um einen Metadatenblock aus mehreren Teilen, wie z.B. dem *compiler* oder den *nodes*. Diese einzelnen Teile beschreiben eine Rechnerumgebung hinreichend.

[90] CodeMeta wird ohne Präfix genutzt, da es kein direkt verwendbares XSD-Schema gibt. Stattdessen nutzt CodeMeta die JSON-LD-Formalisierung.

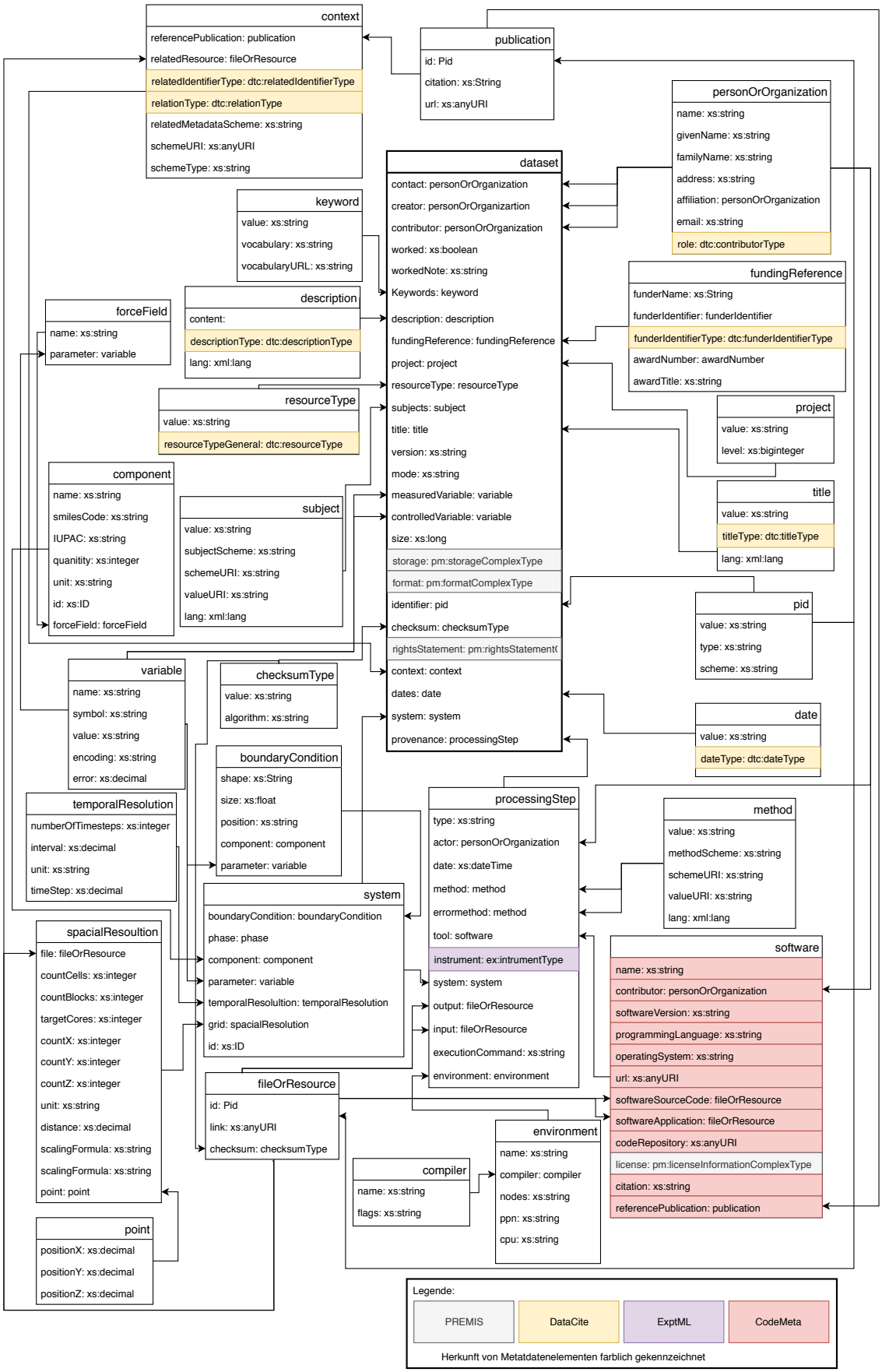

Abbildung 5.7: Das Metadatenmodell *EngMeta* für die Beschreibung von Forschungsdaten. Darstellung mit den XML-Wertebereichen und farbliche Kennzeichnung einzelner Attribute, sofern sie von bestehenden Metadatenstandards stammen. Keine farbliche Kennzeichnung bedeutet Eigenentwicklung.

Metadatenblock/-element	Semantik
worked	Flag für erfolgreichen Simulationslauf
project	Die zugehörigen Projekte
variables	Kontrollierte und gemessene Variablen
environment	Beschreibung der Rechnerumgebung (z.B. Compiler, Prozessoren, verwendete Software, etc.)
spacialResolution	Räumliche Auflösung des beobachteten Systems
temporalResolution	Zeitliche Auflösung des beobachteten Systems
component	Thermodynamische Komponenten (z.B. *Kraftfelder*)
boundaryCondition	Grenzflächenspezifika
method	Spezifische Simulationsmethode (z.B. *Numerisch*)
system	Informationen über das simulierte Zielsystem

Tabelle 5.2: In EngMeta eingeführte, fachspezifische Metadatenblöcke und deren jeweilige Semantik.

5.2.3 Vom Metadatenmodell zum XML-Schema

Die abstrakte Beschreibung der Metadaten wurde aus Gründen der Lesbarkeit und der Nutzbarkeit in eine formale Beschreibung in einer Auszeichnungssprache überführt. Nur in einer formalen Auszeichnungssprache ist das Metadatenmodell für Außenstehende eindeutig verständlich, da ihr eine Konvention inhärent ist, wie die Ausprägungen zu verstehen sind. Ebenso ist das Modell nur von Nutzen, wenn es maschinenles- und -verarbeitbar ist, was durch eine Auszeichnungssprache garantiert wird. Die Entscheidung zugunsten von XML-Schema (XSD) wurde getroffen, weil es eine strikte Validierung der Ausprägungen gegen das Schema ermöglicht und in der Definition in der XSD-Datei jegliche Information vorhanden ist, die für die Maschinenlesbarkeit nötig ist. Ebenso ermöglicht die Definition der Struktur in einem XSD auch die Festlegung der Wertebereiche der einzelnen Attribute in Form von primitiven oder komplexen Datentypen oder kontrollierten Vokabularen. Das komplette XML-Schema findet sich online veröffentlicht[91].

Da es sich um ein datenzentrisches Beschreibungsmodell handelt, wie im vorherigen Abschnitt 5.2.1 bereits begründet wird, bildet die Entität *dataset* die oberste Hierarchiestufe in der Schemadefinition, was in Listing 5.1 dargestellt ist. Alle weiteren Entitäten gliedern sich hierarchisch unterhalb dieser Entität. In diesem Listing ist auch die Nutzung der Metadatenstandards DataCite, PREMIS und ExptML zu sehen. Diese werden in Listing 5.1 in den Zeilen 4-15 eingebunden.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xs:schema
4     xmlns:xs="http://www.w3.org/2001/XMLSchema"
5     xmlns:dtc="http://datacite.org/schema/kernel-4"
6     xmlns:pm="http://www.loc.gov/premis/v3"
7     xmlns:ex="urn:exptml:schema:draft:0.5"
```

[91] https://www.ub.uni-stuttgart.de/engmeta, Zugriff 6.5.2019. In der vorliegenden Arbeit wird sich auf Version 1.1 bezogen.

```
8     >
9   <!--      xmlns="http://www.ub.uni-stuttgart.de/dipling"
10        targetNamespace="http://www.ub.uni-stuttgart.de/dipling"-->
11
12  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
      schemaLocation="http://www.w3.org/2009/01/xml.xsd"/>
13  <xs:import namespace="http://datacite.org/schema/kernel-4"
      schemaLocation="import/datacite-4.0_metadata.xsd" />
14  <xs:import namespace="http://www.loc.gov/premis/v3"
      schemaLocation="import/premis.xsd" />
15  <xs:import namespace="urn:exptml:schema:draft:0.5" schemaLocation
      ="import/exptml_instrument.xsd" />
16
17
18  <xs:element name="dataset">
19    <xs:complexType>
20      <xs:sequence>
21  <!-- descriptive metadata -->
22  <!-- Contact -->
23  <xs:element name="contact" minOccurs="1" maxOccurs="unbounded"
      type="personOrOrganization">
24    <xs:annotation>
25      <xs:documentation>Name and contact information of a person or
       organization that can handle requiries about the data over a
      long time period.</xs:documentation>
26    </xs:annotation>
27  </xs:element>
28
29  [..]
```

Listing 5.1: Kopfzeilen der Schemadefinition von *EngMeta* in XSD. Dabei ist *Data Set* das Wurzelelement (siehe Zeile 18.), unter dem sich alle weiteren Entitäten gliedern.

Schließlich werden alle weiteren Entitäten hierarchisch unter dem *dataset* angeordnet, denen ihrerseits wieder weitere Entitäten untergeordnet sind und die verschachtelt sein können. Das in den Abbildungen 5.4 und 5.5 gezeigte Objekt *Rechnerumgebung*, welches durch CPUs, Compiler usw. beschrieben wird, ist nun in Listing 5.2 als Teil der XML-Schemadefinition dargestellt. Es bildet einen XML *complexType*, welcher ab Zeile 6 eine Auflistung aller seiner Attribute enthält, wobei ein Attribut wiederum selbst ein komplexer Datentyp ist, nämlich der Compiler, der von Zeile 8-14 beschrieben wird.

```
1  <xs:complexType name="environment">
2      <xs:annotation>
```

```
3       <xs:documentation>Computation environment of the data.
    Important for numerical simulations or other software code. Is
    specified by a name, the number of nodes and the number of
    processors per node (ppn). An unbounded number of compilers can
    be specified by name and flags
4       </xs:documentation>
5     </xs:annotation>
6     <xs:sequence>
7       <xs:element name="name" type="xs:string" minOccurs="0"/>
8       <xs:element name="compiler" minOccurs="0" maxOccurs="
    unbounded">
9         <xs:complexType>
10          <xs:all>
11            <xs:element name="name" type="xs:string" minOccurs="1"/
    >
12            <xs:element name="flags" type="xs:string" minOccurs="1"
    />
13          </xs:all>
14        </xs:complexType>
15      </xs:element>
16      <xs:element name="nodes" type="xs:string" minOccurs="0"/>
17      <xs:element name="ppn" type="xs:string" minOccurs="0"/>
18    <xs:element name="cpu" type="xs:string" minOccurs="0" maxOccurs
    ="unbounded" /><!-- new in version 1.1 -->
19    </xs:sequence>
20  </xs:complexType>
```

Listing 5.2: Beispiel des *environment*-Typs in XSD.

Dieses Environment ist selbst Teil der Entität *processingStep*, welcher einen Verarbeitungsschritt repräsentiert, in dem Daten erzeugt werden. Der *processingStep* ist der Entität *provenance* logisch untergeordnet, welche selbst direkt auf der Ebene unterhalb des *dataset* angeordnet ist. Dabei repräsentiert *provenance* die Metadaten über die Herkunft der Daten, also die Prozessmetadaten. Einem Metadatensatz können mehrere Prozessschritte mit jeweils unterschiedlichen Eingabe- und Ausgabedateien (definiert durch *input* und *output*-Ressource unterhalb des *processingStep*) zugeordnet werden. Diese Verschachtelung ist beispielhaft in Abbildung 5.8 für die Kaskade *dataset - provenance - processingStep - environment- compiler* dargestellt. Die weiteren Elemente werden, entsprechend der vorgestellten Methodik, aus dem Objektmodell heraus in das XSD übertragen, um eine hierarchische Form zu erreichen.

5.2.4 Vom XML-Schema zur XML-Ausprägung

Das in den vorherigen Abschnitten erarbeitete XML-Schema steht nun als XML-Schema Definition (XSD) zur Beschreibung von Forschungsdaten zur Verfügung. Wird ein XML-Dokument erstellt, welches den Definitionen des Schemas entspricht, handelt

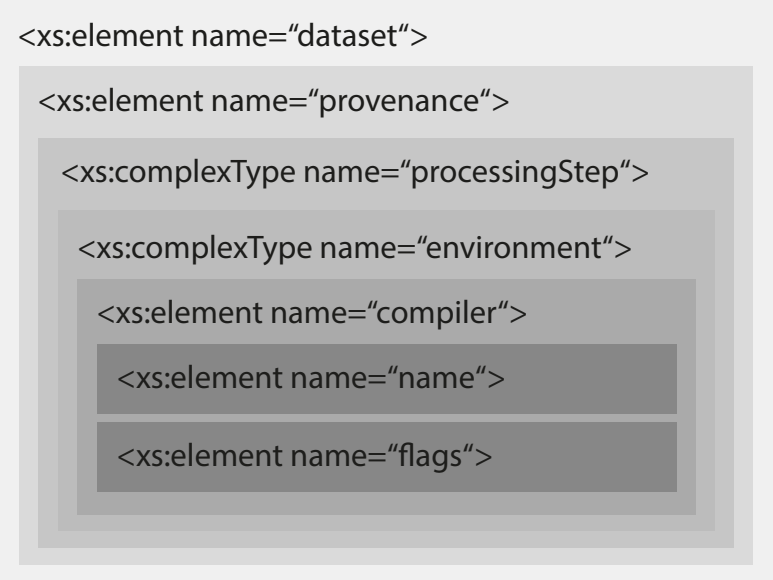

Abbildung 5.8: Darstellung der hierarchischen Ordnung im XML-Schema. Dem Wurzelknoten *Data Set* sind die einzelnen Entitäten untergeordnet, wie z.B. *provenance*, die wiederum die Entität *ProcessingStep* enthält. Dieser besteht aus dem *environment*, welches z.B. aus dem *compiler* besteht, der aus den Basiselementen *name* und *flags* besteht.

es sich um eine Ausprägung des Schemas, die einen konkreten Datensatz repräsentiert. Ein Teil einer Beispielausprägung eines Datensatzes im schemakonformen XML-Format findet sich in Listing 5.3. Dieses zeigt die Informationen zur konkreten Rechnerumgebung, also der Entität *environment*, die innerhalb der *provenance*-Entität die Rechnerumgebung des Datensatzes beschreibt. Dabei kann eine Ausprägung manuell ausgefüllt werden, was allerdings in der Praxis aufgrund der Fehleranfälligkeit und des Aufwands selten vorkommen wird. Typischerweise werden die Metadaten über eine Maske eingegeben oder automatisch erfasst. Wie eine automatische Metadatenerfassung aufgebaut sein muss und funktioniert, ist das Thema das nächsten Kapitels 5.3.

```
1  <?xml version="1.0" encoding="UTF-8" ?>
2  <dataset xmlns:pm="http://www.loc.gov/premis/v3" xmlns:ex="
       urn:exptml:schema:draft:0.5"> <!-- xmlns="http://www.ub.uni-
       stuttgart.de/dipling"> -->
3
4  [..]
5
6   <provenance>
7     <step order="0">
8
9  [..]
10      <environment>
11        <name>bwUniCluster</name>
```

```
12        <compiler>
13          <name>/opt/bwhpc/common/compiler/gnu/7.1.0/bin/gcc GNU
      7.1.0</name>
14          <flags>-mavx       -O3 -DNDEBUG -funroll-all-loops -fexcess-
      precision=fast</flags>
15
16        </compiler>
17        <nodes>10</nodes>
18        <ppn>2</ppn>
19        <cpu>Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz</cpu>
20      </environment>
21    </step>
22  [..]
```

Listing 5.3: Beispiel für eine Ausprägung des Schemas. Dieses repräsentiert einen konkreten Datensatz. Dargestellt ist nur die Rechnerumgebung *environment*, die unterhalb der Entität *provenance* den Datensatz charakterisiert.

5.3 Automatisierte Metadatenerfassung für *EngMeta*

In Kapitel 1.3.2 wurden organisatorische Herausforderungen zur Metadatenerfassung thematisiert. Hier stand neben fehlenden Anreizen auch der Zusatzaufwand für Forschungsdatenmanagement im Mittelpunkt, welcher speziell aus der manuellen Metadatenannotation herrührt. Gemäß (Edwards u. a., 2011) ist diese „Metadata Friction“ schädlich für die Nutzung von Metadatenprodukten, woraus die, in Kapitel 4 formulierte, Forderung entsteht, dass Metadaten nur mit Prozessen zusammen einen Mehrwert bieten können. Damit die „Metadata Friction“ reduziert werden kann, ist die Metadatenannotation möglichst automatisiert zu gestalten. Dies wird nun in diesem Unterkapitel in einen konkreten Entwurf umgesetzt, der auf dem Metadatenmodell *EngMeta* aufbaut und nach Möglichkeit Metadaten automatisch erfasst, wobei die Erfassung nah am Arbeitsprozess ansetzen soll. Zwar setzt der in Kapitel 2.5.1 diskutierte Ansatz iCurate ebenso im simulationswissenschaftlichen Arbeitsprozess direkt nach bzw. bei der Erzeugung der Forschungsdaten an, geht dabei allerdings nicht weit genug. Der Ansatz beschränkt sich auf die Erfassung von vorher annotierten Job-Dateien, und erstreckt sich nicht, wie der Ansatz in der vorliegenden Dissertation, auch auf die Ausgabe- und Logdateien der Simulationscodes selbst. Die automatisierte Metadatenerfassung wird hier anhand des Metadatenmodells *EngMeta* entwickelt, ist aber insofern generisch, als sie dieses Modell nicht voraussetzt. Die Metadatenerfassung selbst ist so konstruiert, dass sich damit jegliche Metadaten, die der Form <*Key*> <*Delimiter*> <*Value*> entsprechen, unabhängig von einem Metadatenzielmodell extrahieren lassen. Dies wird in Unterkapitel 5.3.2 noch näher erklärt.

5.3.1 Erfassbarkeit von Metadaten

Um eine automatisierte Metadatenerfassung zu entwerfen, muss zunächst festgestellt werden, welche der Informationen überhaupt ohne menschliches Zutun durch die Simulationscodes selbst verfügbar sind und sich somit potentiell automatisiert erfassen lassen. Grundsätzlich handelt es sich bei automatisiert erfassbaren Metadaten eher um technische, Informationen über die Rechnerumgebung oder Metadaten, die die Simulationscodes in Logdateien herausschreiben – d.h. alle Informationen, die vom Simulationscode oder der Rechnerumgebung selbst produziert werden. Diese sind in der Regel textuell verfügbar, je nach Simulationsprogramm allerdings in unterschiedlicher Qualität und unterschiedlicher Codierung. Deskriptive Metadaten sind hierbei kaum verfügbar, da sie den Datensatz von einem höheren Standpunkt aus beschreiben. Bezogen auf die vier Metadatenkategorien aus dem vorherigen Kapitel ergeben sich folgende Charakteristika bzgl. der Verfügbarkeit von Informationen.

- Technische Metadaten sind prinzipiell ohne manuelles Zutun verfügbar und sehr gut erfassbar. Alle technischen Eigenschaften, wie z.B. die Dateigrößen, können direkt aus den Dateieigenschaften, die mittels des Dateisystems (typischerweise POSIX) bereit stehen, ausgelesen und teilweise explizit (wie Dateinamen und -größen) oder zumindest implizit (wie Checksummen) erfasst werden.

- Prozessmetadaten sind teilweise verfügbar und gut automatisiert erfassbar. Diese Entstehungsinformationen stehen in vielen Logdateien der Simulationscodes über Umgebungsvariablen oder durch Dateieigenschaften zur automatisierten Erfassung bereit. Allerdings gilt dies nicht für alle Informationen. So sind die Informationen wie die Simulationsmethode oder die beteiligten Akteure nicht maschinell erzeugt und somit auch nicht voll automatisiert erfassbar.

- Fachspezifische Metadaten sind teilweise durch die Simulationscodes verfügbar und somit teilweise erfassbar. Dies hängt stark von der Ausgabe der jeweiligen Simulationscodes ab. Typischerweise werden aber Informationen über die kontrollierten Variablen, die räumliche und zeitliche Auflösung der Simulation und andere simulationscodespezifische Eigenschaften herausgeschrieben. Hier gibt es die meisten Möglichkeiten zur Intervention der simulierenden Wissenschaftlerinnen und Wissenschaftler. Handelt es sich um einen selbst geschriebenen Code, können die Ausgaben direkt beeinflusst und viele Informationen herausgeschrieben werden. Ist der Simulationscode konfigurierbar, kann evtl. Einfluss auf die Art der Ausgabe genommen werden.

- Deskriptive Metadaten sind prinzipiell nicht maschinengeneriert verfügbar und somit nicht automatisiert erfassbar, da sie die Daten von einem höheren logischen Standpunkt aus beschreiben. Kein Simulationscode kann dies leisten – es sind nur die Wissenschaftlerinnen und Wissenschaftler selbst, die ihre Simulationen beschreibend kommentieren können.

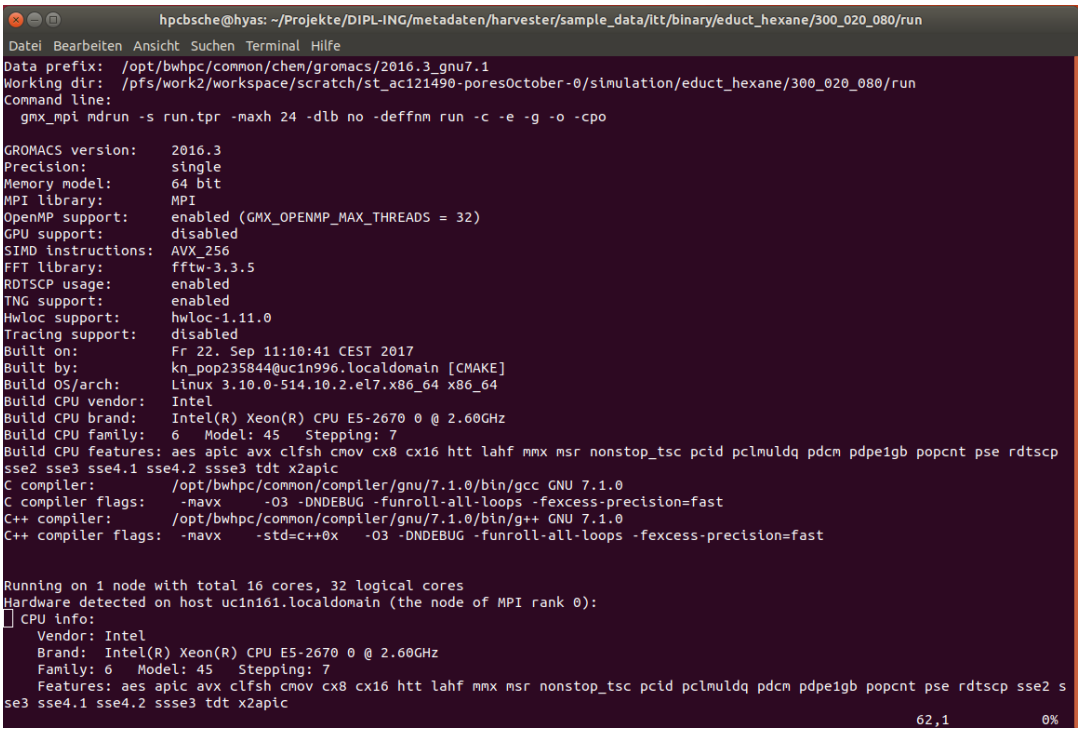

Abbildung 5.9: Ausschnitt einer GROMACS Logdatei, die Informationen über die Rechnerumgebung und Software zeigt.

Es lässt sich also feststellen: Je (fach-)spezifischer die Informationen sind, desto häufiger stehen sie schon durch Simulationscodes generiert zur Verfügung und sind somit gut automatisiert erfassbar. Eine automatisierte Erfassung von Daten kann nur für maschinengenerierte Daten durchgeführt werden, wie z.B. Informationen in Logdateien der Simulationscodes. Ein Ausschnitt einer von GROMACS generierten Logdatei findet sich in Abbildung 5.9. Dort sind viele Informationen zur Rechnerumgebung und Software vorhanden, wie z.B. die die GROMACS-Version, Informationen über die Compiler oder die CPUs. Allerdings ähneln sich viele der deskriptiven Informationen über die Simulationsläufe hinweg, die sich dann beispielsweise in Templates abbildbar hinterlegen und erfassen lassen.

5.3.2 Architektur und Implementierung

Wie sich die Metadaten möglichst geschickt erfassen lassen, spiegelt sich in einem konkreten Softwareentwurf wider, der im Folgenden dargestellt ist. Dabei soll die Software die, auf einem Dateisystem innerhalb des Arbeitsverzeichnisses liegenden, Eingabe- und Ausgabedateien der Simulationscodes nach Metadaten durchsuchen. Die Informationen sind aber über viele Dateien verteilt oder liegen in Form von Dateiattributen bereit. Diese Informationen müssen gesammelt, ausgewertet und schließlich in die Form des *EngMeta*-Metadatenschemas gebracht werden, welches als Ausgabe in einer erzeugten XML-Datei beispielsweise im Verzeichnis abgelegt werden kann. Dieser Zusammenhang ist in Abbildung 5.10 dargestellt und bildet auch gleichzeitig die Grundlegung der Softwarearchitektur. Die Erfassung muss

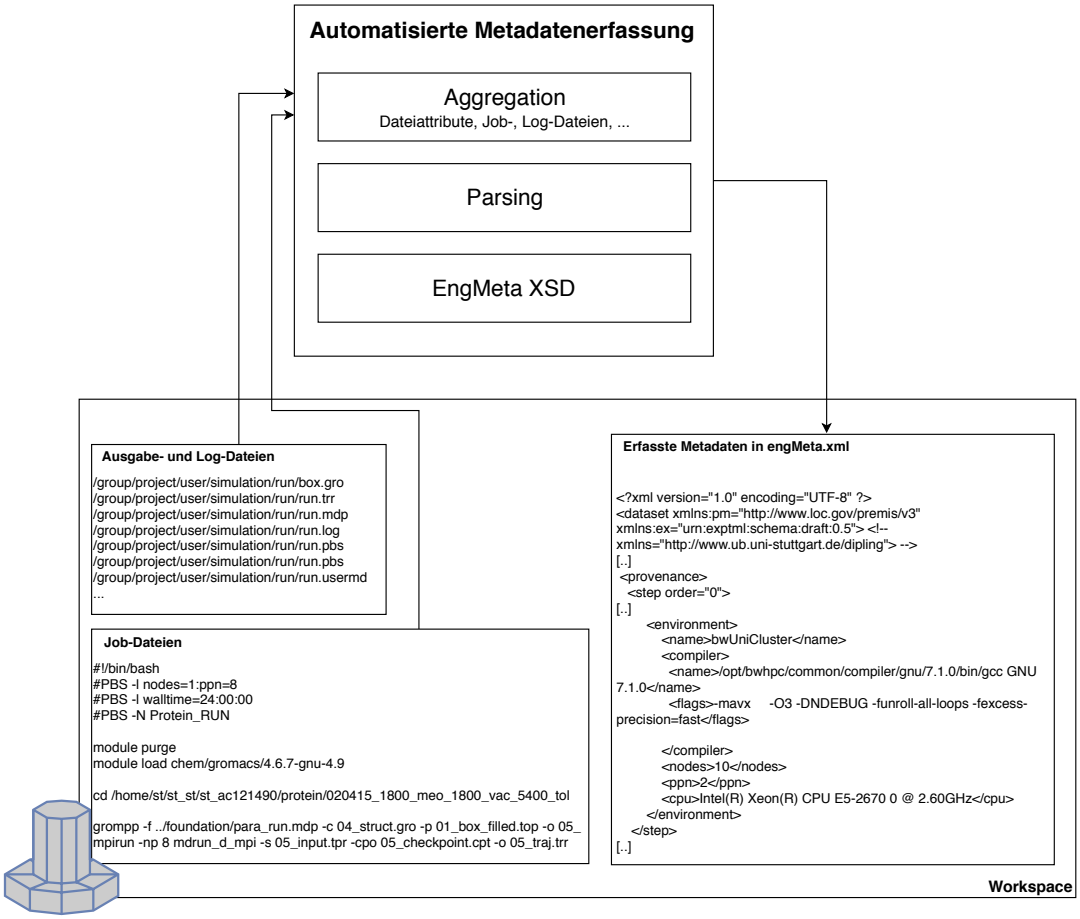

Abbildung 5.10: Architektur der automatisierten Metadatenerfassung.

möglichst generisch – für viele Simulationscodes – anwendbar sein. Dies wird im vorliegenden Ansatz die Universalität gegenüber spezifischen Simulationscodes durch eine Definition der zu erfassenden Metadaten außerhalb des Programmcodes in Form einer Konfigurationsdatei erreicht. In dieser Datei entspricht jede Zeile einer Erfassungsanfrage nach einer Metainformation (entsprechend *EngMeta*). Jede Zeile entspricht einer einfachen Struktur mit den fünf Feldern *metadataKey*, *filename*, *searchKey*, *delimiter* und *semantics*, die jeweils durch Kommata getrennt sind. Die Syntax dieser Datei ist in Listing 5.4 abgebildet, wobei die Bedeutung jedes dieser Felder in Tabelle 5.3 dargestellt ist.

```
1 <metadataKey>,<filename>,<searchKey>,<delimiter>,<semantics>
```

Listing 5.4: Syntax der Konfigurationsdatei

Zwei Beispielzeilen zur automatisierten Erfassung von Compiler-Informationen der Rechnerumgebung, bezogen auf den Simulationscode GROMACS, findet sich in Listing 5.5. Dabei richtet sich der Metadatenschlüssel nach der *EngMeta*-Konvention (siehe auch Kapitel 5.3.3). In Zeile *1* soll der Schlüssel in der Datei mit der Endung *log* unter dem Suchbegriff *C++ compiler* gesucht werden. In der Logdatei wird der

Feldbezeichnung	Bedeutung
metadataKey	Der Metadatenschlüssel, entsprechend dem *EngMeta*-Schema.
filename	Der Name der Datei, in der das Metadatum zu finden ist.
searchKey	Der Suchbegriff, unter dem der Metadatenschlüssel in der Datei zu finden ist.
delimiter	Das Trennzeichen, das den Schlüssel vom Wert des Schlüssels trennt.
semantics	Definiert die Zugehörigkeit von Metadateneigenschaften. Dies ermöglicht Mehrfachvorkommen, z.B. um mehrere kontrollierte Variablen mit jeweils Namen, Symbol, Wert, Encoding und Fehler zu erfassen oder mehrere Compiler.

Tabelle 5.3: Layout der Konfigurationsdatei, in der die zu erfassenden Metadaten festgelegt werden.

Wert durch einen Doppelpunkt vom Suchbegriff getrennt, was im vierten Feld definiert wird. Die *1* im letzten Feld gibt an, dass die beiden Informationen semantisch zusammengehören.

```
1 provenance.processingStep.environment.compiler.name,log,C++
    compiler,:,1
2 provenance.processingStep.environment.compiler.flags,log,flags,:,1
```

Listing 5.5: Eine Ausprägung der Konfigurationsdatei für die Metadatenerfassung von Compilerinformation für den GROMACS-Simulationscode.

Die Erfassungssoftware wurde in Java entwickelt. Die Entscheidung für die Implementierungssprache Java fiel aus Gründen der Interoperabilität, dem Vorhandensein von APIs und der Integration mit XML. Die, durch Bytecode prinzipiell plattformunabhängigen Binärdateien, sollen sich auf vielen Großrechnerplattformen mit verschiedenen unixoiden oder Linux-Betriebssystemen ausführen lassen und außerdem die Option zur Erweiterung der Metadatenerfassung auf Experimentalgeräte offen halten, welche z.B. mit Windows-Messrechnern arbeiten. Die in Java vorhandenen APIs lassen sich produktiv und schnell nutzen und erlauben auch eine parallele Bearbeitung, beispielsweise mit dem Spark-Datenanalyseframework. Auch verfügt die Programmiersprache mit Java Architecture for XML Binding (JAXB)[92] über eine einfache Integrationsmöglichkeit von XML-Schemata und daran anschließend über eine Möglichkeit zur Generierung von XML-Dateien, basierend auf diesem Schema. So wurde für die Implementierung, das als XSD vorliegende, *EngMeta*-Schema mittels JAXB in Java-Klassen konvertiert. Die erzeugten Java-Klassen können dann mit Inhalt gefüllt werden, um nach der eigentlichen Metadatenerfassung eine entsprechende XML-Datei erzeugen zu können. Die Software-Architektur besteht aus der Superklasse $Harvester$, die die beiden Unterklassen $ScannerHarvester$ oder

[92] `https://www.oracle.com/technetwork/articles/javase/index-140168.html`, Zugriff 26.1.2019

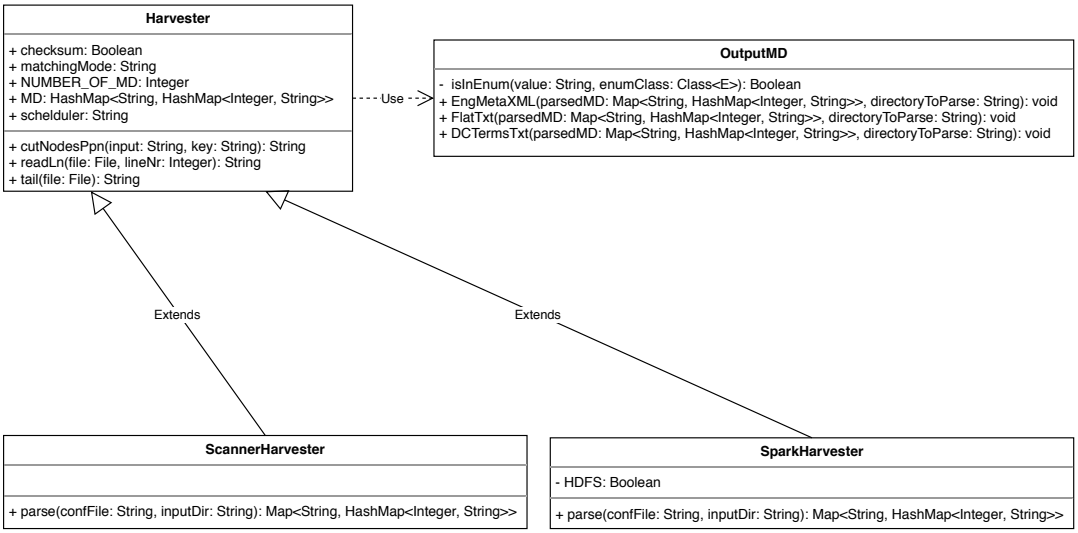

Abbildung 5.11: UML-Klassendiagramm der automatisierten Metadatenerfassung.

$SparkHarvester$ mit den jeweiligen konkreten Erfassungsvarianten (nativ und parallel) erweitern. Daneben findet sich die Ausgabeklasse $OutputMD$, die Methoden enthält, die erfassten Metadaten in verschiedenen Formaten in Dateien zu schreiben und die von der Superklasse genutzt wird. Dieser Zusammenhang findet sich im UML-Diagramm 5.11 dargestellt. In der Softwarearchitektur ist die Erfassung der Metadaten, die in den Klassen $ScannerHarvester$ oder $SparkHarvester$ stattfindet, bewusst unabhängig von der Ausgabe bzw. vom Metadatenmodell implementiert. Dies dient der Adaptierbarkeit der Erfassung, da zunächst unabhängig vom Modell geparst werden kann und sich eigene Metadatenmodelle oder eine Erweiterung von *EngMeta* leicht integrieren lassen. Die interne Datenstruktur, in der die Erfassungsergebnisse verwaltet werden, ist die verschachtelte HashMap MD vom Typ $Map < String, HashMap < Integer, String >>$. Hierbei werden in einer zweidimensionalen Struktur zu jedem, in der Konfigurationsdatei definierten, Metadatenschlüssel alle Suchergebnisse in einer weiteren internen HashMap gehalten, um auch Mehrfachvorkommen von Metadatenschlüsseln abzudecken und sie jeweils zuordnen zu können. Dies bedeutet, dass Werte verschiedener Metadatenschlüssel miteinander in Beziehung stehen und zusammengehören, sofern sie den gleichen Index haben. Ein Beispiel für diese Struktur ist das mehrfache Vorkommen von *contributors* mit den jeweiligen Eigenschaften *name* und *role*, wie in Abbildung 5.13 gezeigt wird. Diese Datenstruktur ist Teil der Superklasse $Harvester$ und ist in Abbildung 5.12 schematisch dargestellt. Der Kern der Metadatenerfassung funktioniert nach dem Prinzip, dass die Konfigurationsdatei Zeile für Zeile gelesen wird. Da jede Zeile definiert, welcher Metadatenschlüssel $MetadataKey$ aus *EngMeta* in welcher Datei mit dem Namen $fileName$ unter welchem Suchschlüssel $SearchKey$ vorkommt und welches Trennzeichen $delimiter$ den Wert vom Schlüssel trennt, kann nun für jeden Metadatenschlüssel die entsprechende Datei geöffnet und nach dem Suchschlüssel durchsucht werden. Wurde dieser gefunden, wird die Zeile in der Datei, in der

metadataKey1	(1, value), (2, value), ... , (n, value)
metadataKey2	(1, value), (2, value), ... , (n, value)

Abbildung 5.12: Schematische Darstellung der HashMap-Datenstruktur.

contributor.name	(1, Mustermann), (2, null), (3, Schembera)
contributor.role	(1, Producer), (2, null), (3, DataManager)

Abbildung 5.13: Beispiel wie die Datenstuktur Mehrfachvorkommen zuordnet.

er vorkommt, extrahiert und am Trennzeichen abgeschnitten. Dieses Suchergebnis wird dann zur globalen Metadatenstruktur MD hinzugefügt, welches die *(Metadatenschlüssel,Wert)*-Paare enthält. Dieses Vorgehen ist in Algorithmus 1 als Pseudocode dargestellt. Die Laufzeitkomplexität der Metadatenerfassung beträgt $\mathcal{O}(m \cdot n)$, wobei m die Anzahl der Zeilen der Konfigurationsdatei ist und n die Anzahl der Zeilen der zu parsenden Dateien. Damit ist die Geschwindigkeit der Metadatenerfassung abhängig von der Größe der zu parsenden Dateien.

Algorithm 1 Hauptschleife der automatisierten Metadatenerfassung.

```
1:  INPUT: configFile : String
2:  MD = newHashMap < String, HashMap < Integer, String >>
3:  while NOT(EOF(configFile)) do
4:    configFileLine[] ⇐ SPLIT(READLINE(configFile), ","))
5:    metadataKey ⇐ configFileLine[0]
6:    fileName ⇐ configFileLine[1]
7:    searchKey ⇐ configFileLine[2]
8:    delimiter ⇐ configFileLine[3]
9:    index ⇐ configFileLine[4] // Semantics field as index
10:   while NOT(EOF(fileName)) do
11:     line ⇐ READLINE(fileName)
12:     if CONTAINS(searchKey, line) then
13:       value ⇐ SPLIT(line, delimiter)
14:       MD ⇐ MD.ADD(metadataKey, (index, value))
15:     end if
16:   end while
17: end while
18: return MD
```

Die Metadatenerfassung wurde in zwei Varianten implementiert, die im Folgenden näher erläutert werden. Beide Varianten werden in Kapitel 6 einer Evaluation unterzogen, die sowohl die Kompatibilität zu mehreren Rechenplattformen als auch die Performance der beiden Varianten auf den Plattformen untersucht.

Native Variante: Java Scanner Die native Variante wurde mit der Java Scanner API implementiert, welche auf Java-Standardbibliotheken aufbaut[93]. Daher funktioniert sie auf allen Rechnersystemen, auf denen eine Java-Laufzeitumgebung verfügbar ist. Diese Variante ist also ohne weitere Voraussetzungen auf diversen Systemumgebungen lauffähig. Dabei richtet sie sich im Wesentlichen nach dem als Pseudocode formulierten Algorithmus 1. Die Scanner API von Java durchläuft dabei die jeweilige, in der Konfigurationsdatei für einen Metadatenschlüssel angegebene, Datei linear und bis zum entsprechenden Suchschlüssel. Zur Speicherung der *Metadatenschlüssel,Wert*-Paare wurde eine assoziative Datenstruktur in Form einer verschachtelten HashMap[94] gewählt, deren Struktur weiter oben beschrieben und in den Abbildungen 5.12 und 5.13 dargestellt ist. Diese Datenstruktur wird nach Beendigung des Erfassungsvorgangs von der entsprechenden Klasse zurückgegeben und kann in der Ausgabeklasse weiterverarbeitet und beispielsweise in XML umgewandelt werden.

Parallele Variante: Spark Im Verlauf der Arbeit zeigte sich, dass die Logdateien der Simulationscodes groß werden können, d.h. schnell im Bereich von Megabytes oder größer liegen. Ebenso soll in den großen Forschungsdaten selbst, sofern diese textuell codiert sind, gesucht werden können. Außerdem soll die Möglichkeit bestehen, viele Forschungsdaten gleichzeitig zu analysieren. Bei diesen Anforderungen kann die Java Scanner API schnell Probleme bzgl. der Performance bekommen, da die Datei linear durchlaufen wird. Daher wurde eine parallele Erfassung, basierend auf dem Spark-Datenanalyse-Framework, implementiert[95]. Dazu mussten die Resilient Distributed Datasets (RDD), auf denen jegliche Datenanalyse mittels Spark basiert, als temporäre Datenstruktur genutzt werden. Auf dieser Datenstruktur müssen die spezifischen Funktionen dieses Frameworks genutzt werden. Die innere Schleife in Algorithmus 1 wird insofern parallelisiert, als dass das zeilenweise Durchlaufen der zu parsenden Datei durch die parallelen Lese-, Filter- und Mapper-Funktionen von Spark ersetzt wird. Sobald mittels der $Filter$-Funktion von Spark (ersetzt in Algorithmus 1 das zeilenweise Lesen durch die innere Schleife in den Zeile 10-16 bzw. Zeile 11) eine Metainformation gefunden wurde, wird sie mit der $Mapper$-Funktion extrahiert (ersetzt in Algorithmus 1 das Auftrennen in Zeile 13). Dann wird sie in diesem Schritt direkt zur globalen Datenstruktur MD hinzugefügt. Insbesondere wird bei Spark auch jede Datei nur einmal am Anfang des Programms in den Speicher geladen und nicht jedes Mal neu durchgegangen, was zu weiteren signifikanten Einsparungen bzgl. Laufzeit und Speicherbedarf führen kann. Der grundsätzliche Nachteil der parallelen Variante ist allerdings, dass auf dem jeweiligen Rechner oder Cluster das Datenanalyse-Framework verfügbar und installiert sein muss.

[93] `https://docs.oracle.com/javase/8/docs/api/java/util/Scanner.html`, Zugriff 26.1.2019.

[94] `https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html`, Zugriff 28.1.2019

[95] `https://spark.apache.org/`, Zugriff 26.1.2019.

```
[hpcbsche@nid00030 .metadata]$ pwd
/mnt/lustre/hpcbsche/itt_data/binary/educt_hexane/300_020_080/run/.metadata
[hpcbsche@nid00030 .metadata]$ ls -alrt
total 20
drwxr-xr-x 2 hpcbsche s29931 4096 Jan 29 15:39 .
-rw-r--r-- 1 hpcbsche s29931 1520 Feb  6 11:46 metadata.txt
-rw-r--r-- 1 hpcbsche s29931 2717 Feb  6 11:46 engMeta.xml
-rw-r--r-- 1 hpcbsche s29931  630 Feb  6 11:46 atom.xml
drwxr-xr-x 3 hpcbsche s29931 4096 Feb 13 11:49 ..
[hpcbsche@nid00030 .metadata]$ tail engMeta.xml
                    <flags>-mavx     -O3 -DNDEBUG -funroll-all-loops -fexcess-precision=fast</flags>
                </compiler>
                <nodes>1</nodes>
                <ppn>8</ppn>
                <cpu>Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz</cpu>
            </environment>
        </step>
    </provenance>
    <size>58</size>
</dataset>
[hpcbsche@nid00030 .metadata]$
```

Abbildung 5.14: *.metadata*-Verzeichnis unterhalb der Simulationsdaten und Ende der XML-Datei mit den erfassten Metadaten nach *EngMeta*-Schema.

5.3.3 Anpassung für verschiedene Simulationscodes

Die Anpassung auf verschiedene Simulationscodes erfolgt ausschließlich über das Schreiben einer Definition in der Konfigurationsdatei. Dabei lassen sich grundsätzlich alle textuell verfügbaren Informationen aus Dateien erfassen, die in der Form von <*Begriff*> <*Delimiter*> <*Wert*> vorliegen, wobei nach dem *Begriff* auch auf Basis von regulären Ausdrücken gesucht werden kann und der *Delimiter* auch z.B ein Leerzeichen sein darf. Ebenso können auch ganze Zeilen oder spezifische Zeilen anhand ihrer Zeilennummer extrahiert werden. Die Wissenschaftlerinnen und Wissenschaftler müssen zur Anpassung wissen, in welcher Datei und unter welchem Begriff ein interessantes Merkmal vorkommt und dies gemäß dem Schema der Konfigurationsdatei auflisten.

Die Definition der Suchschlüssel erfolgte nach den Schlüsseln, wie sie im *EngMeta*-Schema definiert wurden. Dazu mussten die verschachtelten Schlüssel im XSD in eine andere Darstellung gebracht werden. Eine, in der XSD-Repräsentation verschachtelte Definition, wie z.B. die aus Abbildung 5.8, wird wie in Listing 5.6 textuell dargestellt. Jegliche Definition in der Konfigurationsdatei muss sich nach diesen Schlüsselworten richten, soll sie im *EngMeta*-Modell ausgegeben werden.

```
1 provenance.processingStep.environment.compiler.name
2 provenance.processingStep.environment.compiler.flags
```

Listing 5.6: Beispiel für verflachte *EngMeta*-Metadatenschlüssel, bezogen auf Beispiel in Abbildung 5.8.

5.3.4 Integration in den Arbeitsprozess

Da eine der größten Herausforderungen für Forschungsdatenmanagement die Hemmschwelle zur Nutzung von Tools und Metadatenannotation darstellt, müssen die Tools so integrierbar sein, dass sie möglichst wenig Zusatzaufwand bedeuten.

Dateiname	Beschreibung
EngMeta.xml	In dieser Datei finden sich die Ergebnisse des Erfassungsprozesses in der Form des *EngMeta*-Metadatenschemas dargestellt.
atom.xml	Diese Datei enthält alle erfassten Metadaten in Form des DublinCore-Standards und richtet sich nach dem Atom-Format. Dies wird benötigt, um eine SWORD-API anzusprechen, wie z.B. die von Dataverse.
metadata.txt	Diese Datei enthält eine nicht hierarchische Auflistung aller erfassten Metadaten nach der Form *Schlüssel,Wert*, wobei sich der Schlüssel nach dem jeweilig definierten Schlüssel in der Konfigurationsdatei richtet. Diese Form dient den Nutzerinnen und Nutzern zur Weiterverarbeitung der Daten in eigenen Programmen.

Tabelle 5.4: Ausgabedateien des automatisierten Erfassungsprozesses der Metadaten.

Daher ist die Metadatenerfassung portabel gestaltet und, unabhängig von der Repositoriumsanbindung, auch lokal lauffähig. Die Metadatenerfassung bezieht sich dabei immer auf ein Verzeichnis und durchsucht das, in einem Parameter, angegebene Verzeichnis gemäß der Konfigurationsdatei nach Metadaten in den jeweilig spezifizierten Dateien. Ein beispielhafter Aufruf ist in Listing 5.7 dargestellt.

```
1 ./fdm.sh fdm.conf /mnt/lustre/data/educt_hexane/300_020_080/run/
    scanner
```

Listing 5.7: Beispielaufruf der automatisierten Metadatenerfassung.

Nach Beendigung des Erfassungsprozesses wird ein Unterverzeichnis *.metadata* im Simulationsverzeichnis angelegt, welches die Ergebnisse der Erfassung enthält. Die Ergebnisse werden in drei Formaten ausgegeben, die in Tabelle 5.4 dargestellt sind. Eine der Ausgabedateien ist gemäß dem definierten *EngMeta*-Metadatenschema formatiert (*engMeta.xml*), eine weitere gemäß dem DublinCore-Standard (*atom.xml*). Eine zusätzliche Datei ermöglicht die Nutzung der Daten ohne Bezug zu einem Metadatenschema und gibt die Ergebnisse als textuelle Liste aus (*metadata.txt*). Abbildung 5.14 zeigt beispielhaft das *.metadata*-Verzeichnis unterhalb der GROMACS-Simulationsdaten nach der erfolgten Metadatenerfassung mit den drei Ergebnisdateien sowie das Dateiende der *engMeta.xml* mit den erfassten Metadaten nach Schemakonvention. Für die Metadatenextraktion wird für die native Parsingvariante nur eine Shell und Java 1.8 benötigt, die mittlerweile zum Standardumfang in Großrechnerumgebungen gehören. Für die parallele Erfassung ist das Framework Spark nötig, was zusehends auf immer mehr Großrechnerplattformen vorinstalliert ist. Dabei werden die jeweiligen Programme direkt im User-Mode auf den Frontend-Knoten der Großrechner und Cluster ausgeführt. Durch die Definition von Epilog-Skripten lässt sich die Metadatenerfassung direkt an den Simulationsprozess anschließend ohne manuelles Zutun der Wissenschaftlerinnen und Wissenschaftler durchführen, was in Kapitel 6.4 evaluiert wird.

5.4 Der *Scientific Data Officer* als spezifischer Datenkurator im HPC

Wie in (Edwards u. a., 2011) hergeleitet, sind Metadaten eine Kommunikationsform. Sie lassen sich als Vermittlungsprozess zwischen dem/der Datenproduzenten/-in und dem/der Datennutzer/in, verstehen. Allerdings sind einzelne Metadatenprodukte ohne die entsprechenden unterstützenden organisatorischen Prozesse wertlos. Demnach muss eine technische Verbesserung immer mit der Implementierung von organisatorischen Prozessen einhergehen. Diese Sichtweise wird in (Bruce und Hillmann, 2004) unterstützt:

> „As critical as documentation is to improving overall quality, cultural change may be even more critical. We must encourage the growth of an implementer and aggregator culture that not only supports better documentation practices, but also sees dissemination of training, tools, methodologies and research results as essential." (Bruce und Hillmann, 2004, S. 13)

Fehlende Metadatenannotation und dunkle Daten, die durch De-Registrierung und durch Inaktivität herrühren, können nur mit Verfahren und Prozessen verhindert werden, die sich nicht ausschließlich technisch realisieren lassen. Gerade dunkle Daten durch De-Registrierung und Inaktivität werfen auch ethische Fragen auf, wie z.B. die der Erkenntnis und die der Verantwortung[96]. Diese Prozesse benötigen menschliche Intervention, die beispielsweise auch in (Mattmann, 2013) gefordert wird, da es hierbei um die Übernahme von Verantwortung und die Sichtbarmachung von Daten geht. Diese Prozesse finden sich in der Rolle des *Scientific Data Officer* (SDO) wieder. Dieser fungiert als spezifischer Datenkurator für HPC und computergestützte Ingenieurwissenschaften und darüber hinaus überall dort, wo Forschungsdatenmanagement optimiert und dunkle Daten vermieden werden sollen. Dabei kann der SDO auf verschiedenen Ebenen einer Organisationseinheit wirken. Beispielsweise kann die Rolle auf Arbeitsgruppenebene, auf Ebene der Institute und der Universität SDOs geben, die zusammenarbeiten. Seine Rolle ist jedoch nicht an eine Ebene gebunden, da dessen Kompetenzen sowohl technische als auch organisatorische Aufgaben umfassen, die im Folgenden besprochen werden. Dabei wurde der SDO als Rolle bereits vorab in (Schembera und Bönisch, 2017) und in (Schembera und Durán, 2019) publiziert. Sie wird in der Dissertation in den Gesamtzusammenhang von gelingendem Forschungsdatenmanagement gestellt und auf die in den vorherigen Kapiteln ausgearbeiteten Konzepte bezogen.

5.4.1 Technische Verantwortung

Da die fehlende Datendokumentation von Forschungsdaten speziell in den Ingenieurwissenschaften und in anderen Fachbereichen die größte Herausforderung für

[96] Siehe Abschnitt 3.3.

ein geregeltes Forschungsdatenmanagement darstellt, muss der SDO genau hier ansetzen und Metadatenannotation in verschiedener Weise unterstützen. Kenntnisse über Metadatenstandards sind hierfür unabdingbar. Diese Standards müssen bei Bedarf als Anwendungsprofile angepasst oder von Grund auf neu entwickelt werden. Bezogen auf das, in der Dissertation vorgestellte, Metadatenmodell *EngMeta* bedeutet dies insbesondere die Pflege des Modells, welches sich evolutionär entwickelt. Das heißt, dass beispielsweise Fehler behoben und neue Entitäten auf mögliche Integration geprüft werden müssen. Ebenso muss mit den Wissenschaftlerinnen und Wissenschaftlern aus angrenzenden oder ähnlichen Fachbereichen bei Interesse die Übertragbarkeit geprüft werden. Darüber hinaus muss der SDO auch darauf einwirken, dass Metadatenannotation von den Wissenschaftlerinnen und Wissenschaftlern als selbstverständlicher, ergänzender Teil der Forschungstätigkeit gesehen wird, um den Standards guter wissenschaftlicher Praxis zu genügen. Dies kann erreicht werden, indem Werkzeuge wie die automatisierte Metadatenerfassung, propagiert werden. Der SDO muss hierfür beispielsweise auch Anleitung geben, wie die Konfigurationsdateien auf spezifische Simulationscodes angepasst werden oder bei der Erstellung selbst mitwirken.

Ebenso wichtig ist die periodische Überprüfung des Dateninventars nach dunklen Daten. Dies kann aufgrund von Festlegungen geschehen, die vorher mit dem SDO und allen beteiligten Parteien getroffen wurden, so z.B. den Betreiberinnen oder Betreibern des jeweiligen Rechenzentrums und Vertreterinnen und Vertretern der Nutzer selbst. Schließlich muss eine Liste an potentiell dunklen Daten erstellt werden. Mithilfe dieser Liste können dann – sofern vorhanden – die jeweiligen verantwortlichen Personen kontaktiert und das Interesse an den Daten erfragt werden. Dabei richtet sich das Vorgehen nach den, in Kapitel 5.5.3 dargestellten, Verfahren, wobei es teilweise automatisiert werden kann. Dunkle Daten, die durch de-registrierte Nutzerkonten entstehen, können leicht erkannt werden, ebenso wie die durch Inaktivität. Hier kann eine automatisierte Vorauswahl getroffen werden bzw. bei Abmeldung eines Accounts die Zahl der hinterlassenen Daten an den SDO gemeldet werden, der dann direkt das Verfahren in 5.5.3 einleiten kann. Eine periodische Überprüfung sollte mindestens einmal im Halbjahr erfolgen, um ein kontinuierliches Bild der Entwicklung von dunklen Daten zu bekommen.

Zwar handelt es sich bei der Position des SDO nicht um eine rein technische Angelegenheit, allerdings kann auch die Systemadministration der Datenmanagementsysteme von ihm übernommen werden. Der Vorteil, wenn der SDO auch die Systemadministration übernimmt, ist speziell die starke Vereinfachung der periodischen Überprüfung des Dateninventars. Dieser technische Prozess der Datenerhebung kann dadurch wesentlich beschleunigt werden.

5.4.2 Nicht-technische Verantwortung

Wie in den Herausforderungen in Unterkapitel 1.3.2 bereits dargestellt wird, sind die Ausbildungsmaßnahmen entscheidend für ein erfolgreiches Forschungsdatenmanagement und FAIRe Daten. Der SDO muss Ausbildungsmaßnahmen organisieren oder selbst verantworten und als Multiplikator agieren. Im Hinblick auf das Dissertationsvorhaben bedeutet dies beispielsweise die Vermittlung der FAIR-Prinzipien genauso wie die Vermittlung der 7. DFG-Richtlinie. Darüber hinaus müssen die Metadatenprodukte, wie das Metadatenmodell und die -erfassung, bekannt gemacht werden und Schulungen hierüber durchgeführt werden. Dabei sollen die Trainingsmaßnahmen die Arbeit des SDO nicht auf die Wissenschaftlerinnen und Wissenschaftler abwälzen, sondern Prozesse unterstützen, wie insbesondere die Metadatenannotation. Ebenso soll die Kommunikation bzgl. Daten und Forschungsdatenmanagementansätzen in der Position des SDO gebündelt werden und dieser koordinierend wirken. Dies ist sinnvoll, um doppelte Arbeit zu verhindern. Beispielsweise soll hiermit die Entwicklung von Metadatenmodellen kontrolliert und koordiniert werden.

Der SDO muss ebenso die Verantwortung über dunkle Daten übernehmen. Die ethische Rechtfertigung ist, dass so keine Daten verloren gehen können bzw. verloren geglaubte Daten der Öffentlichkeit zur Verfügung gestellt werden können. Ebenso entscheidet der SDO in letzter Instanz (in Absprache mit den erreichbaren Akteuren), in Anlehnung an die, im nächsten Kapitel vorgestellten, Entscheidungskriterien, was mit den ermittelten dunklen Daten geschieht.

5.4.3 Ethische Verantwortung

Die Definition der ethischen Verantwortung des SDO entstammt vor allem der Arbeit mit Juan Durán und wurde im Paper (Schembera und Durán, 2019) veröffentlicht. Obwohl sie nicht das Werk des Autors der Dissertation ist, soll sie hier nachgezeichnet werden, da die ethische Komponente essenziell ist und die Rolle abrundet.

Einerseits ist der SDO, wie oben beschrieben, verantwortlich für das periodische Checken des Dateninventars. Dabei muss er reflektiert handeln, um Missbrauch von normalem Gebrauch zu unterscheiden. Folgender Fall würde beispielsweise von einem automatisiert agierenden Tool erst gar nicht erkannt. Würden Nutzerinnen und Nutzer sich kurz vor der Fünfjahresfrist in das System einloggen, Daten auf dem System umkopieren und so aktiv bleiben, würde ein Tool dieses Verhalten nicht erkennen. Ein SDO kann dieses Verhalten aber als solches erkennen und die oder den entsprechende Nutzerin oder entsprechenden Nutzer kontaktieren. Allerdings obliegt dem SDO auch die Definition, welches Verhalten als problematisch eingeschätzt wird. Beispielsweise kann Kopieren von Daten als schädliches Verhalten eingestuft werden, da dieses weitere Ressourcen blockiert. Der SDO besitzt die Definitions- und Kontrollgewalt über die Nutzerinnen und Nutzer bzw. ihre Daten, welche potentiell auch missbraucht werden kann. Man denke an den Fall, dass der SDO eine bestimmte Gruppe bevorzugt, indem die Daten dieser Gruppe langsamer als gewöhnlich für

dunkel erklärt werden, damit diese Gruppe länger Daten speichern kann. Ebenso müssen die Erzeugung falscher Daten oder die Veränderung bestehender Daten mit in die Überlegungen einbezogen werden. Beide Fehlverhalten führen zu Verfälschung der Datenlage und damit zu veränderter, kontrafaktischer wissenschaftlicher Erkenntnis. Dies betrifft auch und vor allem die Metadaten, da diese die verbindliche Beschreibung der Daten sind. Indem der SDO die Entscheidungsgewalt hat, welche Daten als dunkel gelten, könnte er sehr leicht Daten verschwinden lassen, was zum obigen Punkt der Datenverfälschung gehört. Er kann damit auch implizit die Ausrichtung des HPC-Zentrums bestimmen, indem er vorgibt, welche Daten erneut gerechnet werden müssen.

All diese Gründe rechtfertigen eine Beschränkung und Kontrolle der Macht und der Überwachung des SDO. Eine Möglichkeit zur Kontrolle seiner Tätigkeit ist die periodische Überprüfung seiner Tätigkeit durch die Wissenschaftlerinnen und Wissenschaftler selbst. Ebenso müssen die Wissenschaftlerinnen und Wissenschaftler in Entscheidungsprozesse, die die Daten betreffen, von vornherein eingebunden sein. Da Simulation starke Implikationen für die Gesellschaft haben kann, ist die Forderung nach einer Ethik bzw. einem Verhaltenskodex wichtig. Tuncer Ören schlägt einen simulationsspezifischen Ansatz[97] vor (Ören, 2002), der auch für den SDO gelten muss, da die Daten das Erzeugnis von Simulation und die Basis für wissenschaftliche Erkenntnis sowie politische Entscheidungen sind. Durch die neue DSGVO kann die Macht des SDO (zumindest in Europa) von vornherein durch das Verfahrensverzeichnis geregelt und beschränkt werden. Dieses sollte für jegliche Datenverantwortlichkeiten klar vorhanden und definiert sein.

5.4.4 Abgrenzung zu ähnlichen Rollen

Zunächst scheint es, als wäre der SDO nur eine Abwandlung von schon bestehenden Rollen, wie der Rolle der Systemadministratorin oder des Systemadministrators oder dem Posten des Chief Data Officers (CDO). Im Folgenden wird diskutiert, inwiefern diese Rollen sich von der hier eingeführten Rolle des SDO unterscheiden.

Dem Systemadministrator bzw. der Systemadministratorin untersteht die technische Verantwortung über Computersysteme in einer Institution. Dies bedeutet im Speziellen die Verantwortung über die Rechner- bzw. Speichersysteme und umfasst auch die Installations- und Wartungsarbeiten sowie die Neukonzeption von Hard- und Software in einem Rechenzentrum. Eine Systemadministratorin oder ein Systemadministrator hat hierbei allerdings gerade die gegensätzliche Aufgabe eines SDO, nämlich Systeme – abgesehen von den Schnittstellen – unsichtbar zu machen. Systemadministratorinnen und -administratoren sind nur sichtbar, wenn Probleme auftreten, ansonsten operieren sie im Hintergrund. Im Gegensatz dazu soll der SDO sichtbar als Ansprechpartner und Multiplikator fungieren. Seine explizite Aufgabe

[97] `http://scs.org/ethics/`, Zugriff 9.3.2019.

ist die Sichtbarmachung von Daten und die Verminderung von dunklen, d.h. unsichtbaren Daten. Dabei arbeiten SDO und Systemadministrator Hand in Hand, z.B. beim Sichten des Dateninventars, und können auch identische Personen sein. Bezogen auf das Dissertationsprojekt bedeutet dies, dass der SDO Metadatenprodukte und -dienste wie *EngMeta* oder die automatisierte Erfassung sichtbar machen muss und Systeme so gestalten sollte, dass Metadatenprodukte als ein selbstverständlicher Teil gesehen und genutzt werden.

Eine Position, die zusehends an Bekanntheit gewinnt, ist die des Chief Data Officers (CDO) (Lee u. a., 2014). Der CDO beschäftigt sich mit der Datenbeschaffung, Datenhaltung und Datenanalyse im Geschäftsumfeld. Dabei soll es eine weitere Rolle neben dem Chief Information Officer (CIO), Chief Technology Officer (CTO) und weiteren geben, die die Rolle der Daten und nicht der Systeme (CIO) oder der Forschung und Entwicklung (CTO) fokussiert. Die Rolle ist dabei hierarchisch auf der Managementebene angesiedelt, wohingegen die vorgeschlagene Rolle des SDO parallel neben den Strukturen steht und sich in allen Hierarchieebenen wiederfinden kann. Darüber hinaus geht der CDO in der Konzeption von (Lee u. a., 2014) nicht auf dunkle Daten ein – dieses Problem wird nicht thematisiert.

Eine der Hauptaufgaben des SDO, nämlich die Unterstützung der Metadatenannotation und die Anpassung von Metadatenprofilen auf die Bedürfnisse von Arbeitsgruppen, findet sich bis dato weder in der Verantwortung des klassischen Systemadministrators noch in der des Managements. Die Kenntnisse in Metadatenstandards, -profilen und -produkten wie *EngMeta* sind ein Alleinstellungsmerkmal des SDOs und der kritische Punkt für die Zukunftsfähigkeit von Forschungsdatenmanagement.

Der SDO entspricht der Form nach eher dem Security Officer. Security Officer sind Ansprechpartner in allen Belangen der Cybersicherheit und damit im Gegensatz zum Systemadministrator sichtbar. Sie vereinen Wissen technischer, organisatorischer und rechtlicher Art, machen auf Sicherheitsprobleme aufmerksam, schließen Lücken und agieren als Multiplikatoren (Whitten, 2008). Auf der technischen Ebene muss ein Security Officer beispielsweise unerlaubte Nutzung ermitteln oder Sicherheitslücken schließen. Darüber hinaus fällt Desaster Recovery in seine Verantwortung sowie die Einleitung rechtlicher Schritte gegenüber illegalen Handlungen. Dabei ist die Rolle immer sichtbar: als Kontaktperson, als Vermittlerin oder Vermittler mit dem SDO, der Leitung und mit anderen Institutionen. Strukturell sind diese Rollen also ähnlich. Statt unerlaubte Nutzung zu ermitteln, wird vom SDO die Zahl der dunklen Daten ermittelt. Rechtliche Kenntnisse müssen dazu genauso vorhanden sein wie rein fachliche, d.h. beide Positionen agieren multidisziplinär. Ebenso sind beide Positionen sichtbar innerhalb der Institution, da es bei beiden um die Aufdeckung geht, einerseits von sicherheitsrelevanten Vorfällen, andererseits von dunklen Daten. Sind dunkle Daten durch den SDO ermittelt worden, können durch die im nächsten Kapitel aufgestellten Kriterien verbindliche Entscheidungen getroffen werden.

Abbildung 5.15: Verhältnis zwischen Wichtigkeit und Größe der Daten in den Simulationswissenschaften kategorisiert nach Stufen, adaptiert in (Iglezakis und Schembera, 2018) nach (Reilly u. a., 2011).

5.5 Entscheidungskriterien zur Auswahl und Dauer der Aufbewahrung von Daten

Nachdem im vorherigen Kapitel die Rolle des SDOs eingeführt und gezeigt werden konnte, dass es notwendig ist, Verantwortlichkeit von (dunklen) Daten zu übernehmen, werden nun Entscheidungskriterien ausgearbeitet, die helfen sollen, die Zahl der Daten zu reduzieren. Dies ist aus ressourcenökonomischen Gründen in Anbetracht der Größe von Simulationsdaten wichtig und nötig. Dabei kann die Reduzierung der Datenmenge einerseits über die Dauer der Datenspeicherung und andererseits mittels der Auswahl bzw. Priorisierung von spezifischen Daten geschehen. Hierfür werden die Daten in die Kategorien *Daten als Grundlage für Publikationen*, *sonstige Daten* sowie *dunkle Daten* eingeteilt und jeweils Verfahren erarbeitet und Haltefristen vorgestellt. Diese sollen vom SDO bzw. in verbindlichen Prozessen umgesetzt werden. Insbesondere müssen Kriterien gefunden werden, wann dunkle Daten gelöscht werden können.

5.5.1 Daten als Grundlage für Publikationen

Da im Höchstleistungsrechnen im Allgemeinen und in den computergestützten Ingenieurwissenschaften im Speziellen sehr große Datenmengen anfallen, aus denen auch Publikationen entstehen, ist eine der Anforderungen an das Forschungsdatenmanagement, Entscheidungskriterien vorzuschlagen, ob und wie lange Daten aufbewahrt werden sollten. Die siebte DFG-Empfehlung begründet dies insbesondere mit der wissenschaftlichen Sorgfalt sowie dem Schutz gegenüber Fehlverhalten. Für Daten,

die Grundlage einer Publikation sind, kann zur Konzeption von Entscheidungskriterien die in Abbildung 5.15 dargestellte Datenpyramide eine Orientierung bieten. Diese Pyramide zeigt das Verhältnis zwischen Wichtigkeit und Größe der Daten, die über den kompletten Datenproduktions-, Datenauswertungs- und Publikationsprozess (wie in Kapitel 1.2.2 definiert) angefallen sind, wobei die Wichtigkeit der Daten über den Prozess zunimmt und die Größe der Daten abnimmt.

Ungesichtete Simulationsergebnisse sowie Testläufe und Zwischenstände sind die Grundlage für alle weitere Forschungsarbeit in der Simulation und bilden Stufe 4. Auf der nächsten Ebene, Stufe 3, finden sich ausgewählte Simulationsergebnisse als Basis für die dann im Forschungsprozess folgende Analyse und Auswertung der Daten. Diese Auswertungsergebnisse bilden Stufe 2 und repräsentieren die wichtigen Forschungserkenntnisse, die zusammen mit den Inputfiles, Codes und Skripten bei relativ geringer Größe viele Informationen enthalten, die für die Nachnutzung von Relevanz sind. Auf Stufe 1 findet sich schließlich die Publikation mit den Plots, die aus der Auswertung der Daten auf Stufe 2 entstehen. Dabei wird folgende Orientierung vorgeschlagen, wobei diese unabhängig vom Speicherort der Daten aufgestellt wird:

- Stufe 1: Publikationen mit Plot müssen in jedem Fall aufbewahrt werden, wobei dies in der Regel von Verlagen, Bibliotheken oder Diensten wie `arXiv.org` abgedeckt ist.

- Stufe 2: Aggregierte Daten aus der Datenanalyse, wie z.B. Auswertungsergebnisse sowie Eingabedaten, Startkonfigurationen und die Simulationscodes müssen aufgehoben werden. Sie sind die Basis für wissenschaftliche Publikationen und müssen verfügbar sein, sollte es zu Anfechtungen der wissenschaftlichen Methodik o.Ä. kommen. Diese sollten schon im Sinne von Open Data aufbewahrt werden.

- Stufe 3: Rohdaten und ausgewählte Simulationsergebnisse sollten aufbewahrt werden, da sie eine tiefe Überprüfung möglich machen. Außerdem ermöglichen sie eine erneute Auswertung (z.B. die Suche auf dem gleichen Datensatz nach einem anderen Parameter), ohne die Daten komplett neu generieren zu müssen. Dies ist insbesondere im Sinne der Nachnutzbarkeit der Daten wichtig.

- Stufe 4: Ungesichtete Simulationsergebnisse, Testläufe, Zwischenstände, Restart-Daten sollten nicht über das Projektende hinaus aufbewahrt werden, da sie nur im Simulationsprozess von Bedeutung sind, aber darüber hinaus nur eine geringe Rolle spielen.

Die Dauer der Aufbewahrung richtet sich dabei nach der jeweiligen Stufe und nach der DFG-Richtline. So müssen Daten auf Stufe 1 für immer aufbewahrt werden, was von den Bibliotheken übernommen wird. Daten auf Stufe 2 müssen gemäß DFG mindestens zehn Jahre aufbewahrt werden. Dasselbe gilt für die Daten aus Stufe 3, sofern möglich. Hier können aber je nach verfügbarem Speicher auch kürzere Fristen

gewählt werden. Da die Daten aus Stufe 4 nicht für die Publikation relevant sind, können diese Daten schnell nach Rechenprojektende gelöscht werden oder mit einem Ablaufdatum von wenigen Monaten versehen werden.

5.5.2 Daten ohne Publikation

Daten, aus denen keine Publikation erfolgt ist, müssen gesondert betrachtet werden[98]. Bei dieser Art der Daten muss durch die jeweiligen Wissenschaftlerinnen und Wissenschaftler bewertet werden, ob diese in irgendeiner Form zu einem späteren Zeitpunkt – auch für andere Wissenschaftlerinnen und Wissenschaftler – relevant werden können. Falls dies der Fall ist, sollten hier auch die Daten aus Stufe 3 aufbewahrt werden. Dieser Fall kann eintreten, wenn es gewünscht ist, die Daten, unabhängig von eigenen Publikationen, zur Nachnutzung zur Verfügung zu stellen[99]. Daten, aus denen keine Publikation erfolgt oder erfolgte, werden von der DFG-Empfehlung nicht erfasst, wodurch es keine Vorgaben bzgl. Speicherdauer gibt. Hier wird eine generelle Speicherdauer von fünf Jahren vorgeschlagen – sofern in diesem Zeitraum die Daten niemals benutzt oder angefragt werden, ist das Interesse gering und sie können prinzipiell gelöscht werden. Die Verschiebung der Daten in ein öffentlich zugängliches Repositorium wäre allerdings die erste Wahl, sofern die Größe der Daten nicht dagegen spricht, da mit Thomas Goetz gesprochen werden kann: „[...] your dead end may be another scientist's missing link [...]“ (Goetz, 2007).

5.5.3 Dunkle Daten

Sofern die Existenz dunkler Daten bekannt ist, sind diese nach Möglichkeit zu reduzieren, da dies eine der wesentlichen Aufgaben von Forschungsdatenmanagement darstellt, wie in Kapitel 3.3 dargelegt wurde. Dabei muss bei den Entscheidungskriterien, wie diese Daten zu behandeln sind, zwischen der Art der dunklen Daten unterschieden werden.

Dunkle Daten durch de-registrierte Nutzerinnen und Nutzer

Auf dunkle Daten, die durch de-registrierte Nutzerinnen und Nutzer entstehen, ist technisch kein Zugriff mehr möglich. Diese Art dunkler Daten sollte nach folgendem Verfahren behandelt werden:

1. Prüfen, ob Vertrag/Nutzungsbedingungen den Fall regeln. Falls ja, nach dieser Vereinbarung verfahren.

[98] Diese Daten werden von Thomas Goetz in (Goetz, 2007) als dunkle Daten bezeichnet.

[99] Dieser Modus wurde für die Daten gewählt, die am HLRS im Rahmen des UPSCALE-Projektes 2012 (`http://proj.badc.rl.ac.uk/upscale`) entstanden sind. Hierbei wurden globale Atmosphärensimulation mit dem HadGEM3 Modell in verschiedenen Auflösungen (130 km, 60 km, 25 km) am HLRS auf dem Rechensystem HERMIT gerechnet und die knapp 440 TB Daten zur Nachnutzung auf der JASMIN-Plattform am STFC-CEDA übertragen und bereitgestellt. Auf diesen Daten basiert beispielsweise die Studie (Roberts u. a., 2015), in der das Entstehen von tropischen Zyklonen analysiert wird.

2. Falls kein Vertrag besteht: Ermittlung einer Kontaktperson, z.B. des bzw. der Account-Eigentümer/-in selbst oder deren Instituts/Projektleiterin oder -leiter. Aushandlung über weiteres Vorgehen treffen, z.B. Überantwortung der Daten an andere Nutzerinnen und Nutzer, Sichern der Daten auf institutseigenen Speichersystemen oder Löschung der Daten.

3. Wenn kein Kontakt ermittelbar, sollten die dunklen Daten gelöscht werden oder in die Obhut des SDO genommen werden. Dieser kann die dunklen Daten in ein öffentlich zugängliches Archiv überführen, womit sie nicht mehr als dunkel gelten. Selbstverständlich kann das Löschen von Daten nur im Rahmen der rechtlichen Möglichkeiten erfolgen. Löschen der Daten ist rechtlich nur sicher, wenn dazu in den Nutzungsbedingungen klare Abmachungen getroffen wurden (Lauber-Rönsberg, Krahn und Baumann, 2018).

Grundsätzlich sollten dunkle Daten vermieden werden, indem ein Vertrag im Vorhinein klar regelt, was mit den Daten passiert, wenn ein Account abgemeldet wurde. Sollte es keinen Vertrag geben, können die Daten nach dem obigen Verfahren ohne weitere Haltefristen gelöscht werden, wenn kein Kontakt o.Ä. ermittelbar ist.

Dunkle Daten durch inaktive Nutzerinnen und Nutzer

Dunkle Daten, die durch inaktive Nutzerinnen und Nutzer entstehen, sind trotzdem noch mit einem aktiven Account verknüpft. Auf diesen Account und damit auch auf die Daten selbst ist allerdings schon über eine gewisse Zeitspanne nicht zugegriffen worden, wonach diese als potentiell dunkel gelten können. Hier sollte nach folgendem Verfahren vorgegangen werden:

1. Prüfen, ob Vertrag und/oder Nutzungsbedingungen den Fall regeln. Falls ja, kann nach dieser Vereinbarung verfahren werden.

2. Falls kein Vertrag besteht: Ermittlung der Kontaktdaten des Account (oder entsprechende Instituts/Projektleiterin oder -leiter). Interesse an den Daten abfragen, sofern die Genannten erreichbar sind.

3. Wenn kein Kontakt ermittelbar ist oder kein Interesse an den Daten besteht, können diese dunklen Daten nach dem Ablauf einer weiteren Frist gelöscht werden. Alternativ können diese dunklen Daten dem SDO überantwortet werden und bis zum Projektende aufbewahrt und dann ggf. in einem Repositorium veröffentlicht werden, sofern rechtlich möglich (s.o.). In diesem Fall muss auch thematisiert werden, was mit dem Account geschieht, wenn die Account-Eigentümerin oder der Account-Eigentümer nicht erreichbar ist.

Dieser Fall ähnelt dem Fall der dunklen Daten durch de-registrierte Nutzerinnen und Nutzer, allerdings besteht hier noch der technische Zugriff auf den Account. Die Frage nach der Zeitspanne, ab der Daten ohne Zugriff als dunkel gelten, unterscheidet

sich stark von Speichersystem zu Speichersystem und von Fall zu Fall. Im Falle eines mittelfristigen Auslagerungsspeichers (HSM) kann beispielsweise eine Frist von fünf Jahren gesetzt werden, ab der Daten ohne Zugriffe bei bestehendem Account als dunkel gelten. Diese Frist wurde in Kapitel 3.1.2 hergeleitet. Für Archivspeichersysteme und Repositorien kann eine Frist von zehn Jahren für Daten gewählt werden, aus denen Publikationen hervorgegangen sind. Für andere Daten kann die Frist kürzer angesetzt werden.

Dunkle Daten durch fehlende Metadatenannotation

Auch Daten, denen eine hinreichende Datendokumentation fehlt, können als dunkle Daten gelten. Da diese Art der dunklen Daten keinerlei Aussage über das (evtl. mangelnde) Interesse der Nutzerinnen und Nutzer an den Daten macht, sondern nur ihr (mangelndes) Interesse an Datendokumentation impliziert, können diese Daten auch nicht gelöscht werden. Es muss davon ausgegangen werden, dass mit ihnen aktiv gearbeitet wird, sofern sie nicht zusätzlich in eine der ersten beiden Kategorien der dunklen Daten fallen. Ebenso kann keine Aussage über Haltefristen gemacht werden. Darüber hinaus ist nicht ermittelbar, ob die Metadaten evtl. in einem anderen System verwaltet werden.

Kapitel 6

Evaluation: Anwendung und Verifikation der Konzepte

In diesem Kapitel sollen die in Kapitel 5 erarbeiteten Konzepte validiert werden. Dabei muss die Güte des Metadatenmodells *EngMeta* sowie seine Adaptierbarkeit auf verschiedene Fachdisziplinen und ihre Datenformate geprüft werden. Ebenso muss die Adaptierbarkeit der Metadatenerfassung auf verschiedene Rechnerumgebungen und Simulationscodes sowie ihre Performance evaluiert werden. Darüber hinaus muss die Integration in den Arbeitsprozess getestet werden, die sowohl die Ausführung in einer entsprechenden Systemumgebung für Großrechner umfasst als auch das Einspielen von Daten und erfassten Metadaten in das Dataverse-Repositorium der Universität. Für die Evaluation des *Scientific Data Officer* werden Evaluationsansätze besprochen. Zum Ende des Kapitels wird überprüft, inwiefern durch die Lösungskonzepte die in Kapitel 4 aufgestellten Anforderungen erfüllt werden und dunkle Daten durch einen optimierten Datenlebenszyklus reduziert werden können.

6.1 Evaluationskriterien

In Tabelle 6.1 findet sich ein Überblick über die Art der jeweiligen Evaluation der Konzepte, die in diesem Kapitel vorgenommen wird. Dabei wird das Metadatenmodell nach Güte und Adaptierbarkeit qualitativ bewertet. Die Metadatenerfassung kann nach Korrektheit und Adaptierbarkeit qualitativ, nach Performance auch quantitativ evaluiert werden. Die Integration des Metadatenmodells bzw. des Erfassungstools kann ebenso qualitativ bzgl. der Integration in den Arbeitsprozess und der Integration in das Repositorium (d.h. vollautomatisiertes Einspielen in das Repositorium) beurteilt werden. Es zeigt sich, dass die meisten Konzepte nur qualitativ geprüft werden können, da die statistische Basis von einer Vielzahl an im Repositorium liegenden Daten fehlt. Dies liegt am Projektstadium des DaRUS-Repositoriums, welches noch nicht in den Regelbetrieb überführt wurde. Darüber hinaus werden jeweils die in Kapitel 4 formulierten Anforderungen abgeprüft.

	Kriterium	Bewertung
Metadatenmodell	Güte	qualitativ
	Adaptierbarkeit	qualitativ
Metadatenerfassung	Adaptierbarkeit	qualitativ
	Performance	quantitativ
Integration	Arbeitsprozess	qualitativ
	Repositorium	qualitativ

Tabelle 6.1: Überblick Evaluationskriterien.

6.2 Metadatenmodell

6.2.1 Güte des Metadatenmodells

Die Güte eines Metadatenmodells zu erfassen, ist ein schwieriges Unterfangen, wie in Kapitel 2.2.5 bereits festgestellt wird. Jedoch lassen sich anhand der in (Bruce und Hillmann, 2004) und (NISO, 2007) aufgestellten Forderungen einige qualitative Aussagen über Metadaten treffen, mit denen im Folgenden *EngMeta* qualitativ bewertet werden soll. Dazu sollen zunächst die in (NISO, 2007) aufgestellten Metadatenprinzipien überprüft werden:

1. **Standards**: Dem ersten Metadatenprinzip folgend, soll ein Metadatenmodell einem anerkannten Standard entsprechen oder zumindest (in Teilen) auf diesem basieren, wobei proprietäre oder selbstgebaute Schemata vermieden werden sollten. Dem Metadatenmodellentwurf soll eine Anforderungsanalyse zugrunde liegen.

   Erfüllt. Zwar handelt es sich um ein selbst aufgebautes Metadatenschema, allerdings wurden in einer Anforderungsanalyse die genauen Bedarfe mit Fachwissenschaftlerinnen und Fachwissenschaftlern ermittelt, wie in Kapitel 5.2 dargelegt ist. Schließlich wurde die Schaffung eines eigenen Modells als unabdingbar angesehen, da keines der bestehenden Modelle die Anforderungen erfüllen konnte. Allerdings basiert *EngMeta* – sofern diese verfügbar waren – auf anerkannten, etablierten Standards. So entstammen die deskriptiven Metadaten größtenteils dem DataCite-Standard, um maximale Kompatibilität mit Repositorien und die Möglichkeit zur DOI-Vergabe zu erreichen. Die technischen Metadaten in *EngMeta* richten sich nach dem PREMIS-Standard, die Prozessmetadaten sind kompatibel zu Prov bzw. ProvOne, CodeMeta und ExptML. Die Herkunft der jeweiligen Metadatenfelder findet sich in Kapitel 5.2.2 in Abbildung 5.7 dargestellt und wird im Metadatenschema selbst mit Präfixen angezeigt (*dtc:* für DataCite, *pm:* für PREMIS und *ex:* für ExptML). Alle auf CodeMeta aufbauenden Metadatenfelder werden direkt ohne die Angabe des Namensraums verwendet.

2. **Interoperabilität**: Dem zweiten Metadatenprinzip folgend, sollen gute Metadaten interoperabel gestaltet sein. Dies bedeutet, die Metadaten sollen einfach

zwischen Systemen austauschbar und in einer zentralen Stelle registiert sein. Darüber hinaus bedeutet dies nicht nur technische Interoperabilität, sondern auch Kontextunabhängigkeit – die Daten sollen unabhängig von ihrem Speicherort oder ihrer Produktionsumgebung verstanden werden können.

Erfüllt. Die Anforderung nach Interoperabilität wird durch die Wahl der Formalisierung gewährleistet. Die Metadaten wurden als XML-Schema implementiert, sodass diese maschinenlesbar und systemunabhängig genutzt werden können. Die Registrierbarkeit wird durch die Einspeisung der Daten und Metadaten in das Dataverse-Repositorium gewährleistet, welches auch DOIs für die Daten vergeben kann. Dies wurde lokal an der Universität Stuttgart aufgesetzt und getestet, wie im folgenden Unterkapitel 6.4.2 dargestellt ist. Zur Gewährleistung der Kontextunabhängigkeit stellt das Metadatenmodell eine umfassende Beschreibung dar, die neben technischen und deskriptiven Merkmalen auch fach- und prozessspezifische einschließt und so zu einem übergreifenden Verständnis auf allen diesen Ebenen führt.

3. **Normdateien**: Gute Metadaten basieren – sofern möglich – auf Normdateien und kontrollierten Vokabularen, um Mehrdeutigkeiten zu vermeiden.

   Erfüllt. Dieses Prinzip wird erfüllt, indem die kontrollierten Vokabulare genutzt werden, die auch in den integrierten Standards wie PREMIS, DataCite usw. verwendet werden. Darüber hinaus wurden auch mittels des *<xs:restriction>*-Tags Werte von vornherein beschränkt, nämlich die Datenerzeugungsmethode, der PID-Typ und der Prozessschritt-Typ.

4. **Rechte**: Die Metadaten sollen eindeutige Aussagen zu Nutzungsrechten und sonstigen lizenzrechtlichen Belangen der beschriebenen Daten enthalten.

   Erfüllt. Im Metadatenmodell *EngMeta* ist eigens ein auf PREMIS basierendes *rightsStatement*-Feld enthalten, welches rechtliche Informationen enthält. Dessen Einbindung ist in Listing 6.1 dargestellt und verweist auf den *rightsStatementComplexType* von PREMIS, welcher die Felder *rightsStatementIdentifier*, *rightsBasis*, *copyrightInformation*, *licenseInformation*, *statuteInformation*, *otherRightsInformation*, *rightsGranted*, *linkingObjectIdentifier*, *linkingAgentIdentifier* enthält und somit eine Vielzahl von (lizenz-)rechtlichen Informationen speichert.

```
<xs:element name="rightsStatement" type="
    pm:rightsStatementComplexType" minOccurs="0" maxOccurs="
    unbounded">
```

Listing 6.1: Einbindung eines PREMIS- Datentyps zur Rechtedarstellung in *EngMeta*. Dieser findet sich in (PREMIS, 2015) definiert.

5. **Erhaltungsmetadaten**: Metadaten müssen Langzeitarchivierung und Kuration der Daten ermöglichen, indem technische und Erhaltungsmetadaten enthalten sind.

Erfüllt. Durch die Integration von PREMIS wurden technische Metadaten wie Checksumme, Dateigröße und Dateiformat abgebildet, die wesentlich zur Langzeitarchivierung und -erhaltung beitragen. Zu den Erhaltungsmetadaten gehören im 5. Metadatenprinzip auch Prozess- und Speicherinformationen, die in *EngMeta* mit der *processingStep*-Entität abgebildet werden können.

6. **Meta-Metadata**: Metadaten sollen selbst mit Informationen versehen sein, also „Meta-Metadaten“ enthalten, damit sie für Dritte nachvollziehbar sind.

   Erfüllt. Diese Forderung wird erfüllt, da *EngMeta* als XML-Schemadatei vorliegt und somit eine strikte Definition inhärent ist. Ebenso wurde die in XML vorhandene Möglichkeit zur Dokumentation mittels <*xs:documentation*>-Tags für alle XML-Datentypen genutzt. Darüber hinaus wurden Kommentare genutzt, um weitere Informationen zum Schema zu liefern. Weiterhin findet sich in (Schembera und Iglezakis, 2019) eine Dokumentation.

Im Folgenden wird *EngMeta* einer qualitativen Analyse entlang der in (Bruce und Hillmann, 2004) vorgeschlagenen Kategorien unterzogen:

1. **Vollständigkeit**: Das beschriebene Objekt soll so vollständig wie möglich und wie nötig beschrieben werden. Darüber hinaus sollten alle Ausprägungen so viele Elemente des Modells wie möglich nutzen.

   Teilweise erfüllt/teilweise noch nicht prüfbar. *EngMeta* wurde mit Fachwissenschaftlerinnen und Fachwissenschaftlern entworfen und dabei besonderer Wert auf vollständige Abbildung aller für die Forschungsdaten maßgeblichen Teile des Forschungsprozesses gelegt. Dies ist in der Herleitung des Objektmodells, die in Kapitel 5.2.1 dargestellt wird, expliziert. Wie viele der Metadatenfelder tatsächlich genutzt werden, lässt sich nur quantitativ mit einer größeren Datenbasis ermitteln.

2. **Genauigkeit**: Mehrdeutigkeiten sollten vermieden werden, zumindest muss die gespeicherte Information korrekt sein.

   Erfüllt. Diese Forderung ist für *EngMeta* erfüllt, da im XSD auch die Wertebereiche der jeweiligen Attribute festgelegt und automatisiert abgeprüft werden können. Bei im Modell integrierten Metadatenstandards werden auch kontrollierte Vokabulare genutzt.

3. **Herkunft**: Es sollten Informationen über den Entstehungsprozess auch der Metadaten verfügbar sein.

   Erfüllt. *EngMeta* enthält vielzählige Informationen zur Darstellung des Entstehungsprozesses der Daten und ist auch in der Lage, Informationen über die Metadaten darzustellen. Dies könnte beispielsweise durch die Definition eines oder mehrerer *processingSteps* geschehen, der nur die Metadatenentstehung abbildet.

4. **Übereinstimmung mit den Erwartungen**: Dies betrifft insbesondere die Einbeziehung der Forschungsgemeinschaft, für die das Metadatenmodell entworfen wird, um Missverständnisse zu vermeiden. Hier können auch kontrollierte Vokabulare helfen.

   Erfüllt. Die Community wurde von Beginn an beim Entwurf von *EngMeta* beteiligt. Das Objektmodell wurde in das Metadatenmodell überführt, wodurch alle im Objektmodell definierten Eigenschaften erfasst wurden. Dies ist in Kapitel 5.2.1 dargestellt.

5. **Logische Konsistenz und Kohärenz**: Diese Forderung soll sicherstellen, dass Elemente so definiert werden, dass sie Standards entsprechen oder Standardmethoden beim Metadatenhandling angewandt werden.

   Erfüllt. Die Forderung wird erfüllt, da die Elemente im XSD-Schema überprüfbar sind und das Modell auf den Standards DataCite, PREMIS, ProvONE, ExptML und CodeMeta basiert.

6. **Aktualität**: Aktualität muss durch persistente IDs garantiert werden, die allerdings synchron gehalten werden müssen.

   Teilweise erfüllt/teilweise nicht prüfbar. *EngMeta* enthält Felder zur Speicherung einer PID. Beim Veröffentlichen des Datensatzes in Dataverse bekommen die Daten eine DOI zugewiesen. Das Synchronhalten kann nicht innerhalb des Metadatenmodells erfolgen, sondern muss über externe Dienste oder ein Repositorium garantiert werden und konnte noch nicht getestet werden.

7. **Zugreifbarkeit**: Grenzen (technische, organisatorische, aber vor allem ökonomische und handelsbezogene), die die Zugreifbarkeit der Daten, Metadaten und des Metadatenmodells einschränken, müssen abgebaut werden.

   Erfüllt. Das Metadatenmodell ist allgemein zugänglich und nutzbar und wird über eine Website zur Verfügung gestellt[100]. Ebenso ist *EngMeta* erweiterbar, da die frei verfügbare XSD-Datei ohne Probleme ergänzt werden kann. Die Daten und Metadaten selbst sollen später regulär im Repositorium DaRUS veröffentlicht werden, sofern dies von den Erstellerinnen und Erstellern gewünscht ist.

In der qualitativen Bewertung erfüllt *EngMeta* alle in (Bruce und Hillmann, 2004) und (NISO, 2007) aufgestellten Kriterien. Die einzigen kritischen Kategorien sind die *Aktualität* und *Vollständigkeit* nach (Bruce und Hillmann, 2004). Die Aktualität der Daten kann nicht innerhalb der Metadaten festgestellt werden und das Nachziehen der Daten hinter dem PID ist kein Problem der Metadatenbeschreibung. Die Vollständigkeit der Metadatenbeschreibung kann schlussendlich nur quantitativ anhand der Ausprägungen, also der Metadatensätze selbst, geprüft werden, die noch nicht in ausreichender Zahl vorliegen.

[100] `https://www.ub.uni-stuttgart.de/engmeta`, Zugriff 14.3.2019.

Abgleich mit den Anforderungen An dieser Stelle soll auch überprüft werden, ob die in Kapitel 4 aufgestellten funktionalen Anforderungen bzgl. Metadaten erfüllt werden können. Die funktionale Anforderung FA 1.1 (Metadaten müssen Suchkriterien (sowohl fachspezifisch, als auch allgemein, zur Verfügung stellen) wird erfüllt, da *EngMeta* Metadatenelemente wie Autorschaft, Jahr (allgemein) wie auch Informationen über das simulierte System oder die kontrollierten Variablen enthält, welche beide als Suchkriterien für die Fachwissenschaftlerinnen und Fachwissenschaftler gelten. Die Anforderung FA 1.2 (Metadaten müssen relevante und präzise Attribute enthalten) wird erfüllt, da die Attribute gemeinsam mit den Fachwissenschaftlerinnen und -wissenschaftlern diskutiert und entworfen wurden. Dies garantiert zumindest Relevanz. Präzision wurde erreicht, indem zwei unterschiedliche Disziplinen im Fachbereich beteiligt waren. So wurde der "kleinste gemeinsame Nenner" iterativ herausgearbeitet. Die FA 1.3 (Metadaten müssen Informationen über die Herkunft der Daten enthalten) wird erfüllt, da es die Entitäten *provenance* und *processingStep* gibt, welche detaillierten Aufschluss über die Herkunft und Datenerzeugung geben. Die Anforderung FA 1.4 (Metadaten müssen fach-/domänenspezifische Informationen enthalten) wird erfüllt, da eine Reihe von für die Ingenieurwissenschaften spezifischen Metadatenfelder vorhanden sind, was in Kapitel 5.2.1 in Tabelle 5.2 gezeigt wurde. Ebenso können die Anforderungen FA 1.5 (Metadaten müssen deskriptive Informationen enthalten) und FA 1.6 (Metadaten müssen technische Informationen enthalten) erfüllt werden, da technische und deskriptive Beschreibungsmöglichkeiten im Modell von vornherein vorgesehen wurden und sich direkt auf der obersten Hierarchiestufe in der Entität *dataset* finden.

Für eine statistische Untersuchung der Metadatenqualität, wie sie in Kapitel 2.2.5 diskutiert wurde, fehlt die quantitative Basis. Diese Art der Untersuchung arbeitet mit den Ausprägungen des Modells, also mit ausgefüllten Metadatensätzen. Da momentan nur testweise Daten mit *EngMeta* beschrieben und im *DaRUS*-Repositorium abgelegt wurden, ist die Basis für eine quantitative Studie noch zu gering.

6.2.2 Adaptierbarkeit auf andere Anwendungsfälle

In diesem Unterkapitel soll die Übertragbarkeit von *EngMeta* diskutiert werden. Das Metadatenmodell wurde entlang der Anwendungsfälle Thermodynamik und Aerodynamik entwickelt und deckt insbesondere den Bedarf zur Datendokumentation aus diesen Bereichen ab. Dies wird durch die eingeführten fachspezifischen Metadatenelemente, wie sie in Tabelle 5.2 in Kapitel 5.2.2 dargestellt sind, ermöglicht. Allerdings ist die Nutzung des Metadatenmodells nicht auf diese zwei Anwendungsfälle beschränkt. Prozessmetadaten zur Rechnerumgebung können mittels des *environment* Metadatenelements erfasst werden und sind weder auf Höchstleistungsrechner noch die Ingenieurwissenschaften beschränkt. Innerhalb des *processingStep*-Metadatenelements kann die *instrument*-Entität, welche auf ExptML basiert, auch Daten zu Experimentalgeräten erfassen. Die Metadatenentität *Variable*, die sich aus

name, *value*, *symbol*, *encoding* und *error* zusammensetzt, ist nicht an die obigen Fachgebiete gebunden, sondern universell einsetzbar. Das modellierte Zielsystem wird insofern allgemein ausgedrückt, als es aus den Parametern, gemessenen und kontrollierten Variablen besteht und durch eine räumliche und zeitliche Auflösung der Simulation definiert wird. All diese Elemente sind nicht exklusiv für die Thermo- und Aerodynamik, sondern finden sich in den meisten Modellbildungsprozessen der computergestützten Ingenieurwissenschaften. Damit können alle Wissenschaftszweige, die in ihrer Methodik kontrollierte und gemessene Variablen und Parameter nutzen, *EngMeta* prinzipiell einsetzen. Neben den Wissenschaftsformen, die mit modellbasierter Computersimulation arbeiten, sind dies auch solche mit experimentellem Charakter. Ebenso ist die prozessspezifische Metadatenentität *Software* vorhanden, die universell für Forschungssoftware einsetzbar ist. Die einzigen strikt auf die thermodynamische Simulation zugeschnittenen Metadatenentitäten sind *ForceField* zur Angabe der Kraftfelder, *Component* zur Abbildung der thermodynamischen Komponenten und *Phase*. Diese Informationen könnten auch für andere Fachbereiche, wie z.B. die Chemie, von Interesse sein. Diese Metadatenelemente sind aber keine Pflichtfelder. Das Metadatenmodell kann somit überall dort zur Anwendung kommen, wo mittels Computern, Software und eines Modells eine Simulation durchgeführt wird, die das simulierte System räumlich und zeitlich imitiert.

6.3 Automatisierte Metadatenerfassung

6.3.1 Adaptierbarkeit

Anpassbarkeit an Metadatenmodelle

Die Anpassung der automatisierten Erfassung auf andere Metadatenmodelle als *EngMeta* erfolgt über die Implementierung von Java-Klassen für das jeweilige Metadatenmodell und das Schreiben einer entsprechenden Ausgabemethode. Einem XSD-Metadatenschema entsprechende Java-Klassen lassen sich automatisiert mit dem JAXB-Framework mittels des Kommandozeilen-Tools *xjc* oder innerhalb der *Eclipse IDE* erstellen[101]. Schließlich muss in der Ausgabeklasse *OutputMD.java* definiert werden, wie die interne Datenstruktur *MD* (verschachtelte HashMap) auf die Java-Klassen des Metadatenmodells abgebildet wird, was insbesondere durch die Ausprogrammierung von Setter-Methoden der oben erzeugten Java-Klassen erfolgt[102]. Da die Metadatenerfassung aber auch standardmäßig eine flache Auflistung der geparsten Metadaten in einer Textdatei enthält, kann der Implementierungsaufwand auch in ein Umwandlungswerkzeug investiert werden, welches die flache Aufzählung der Metadaten als *Metadatenschlüssel*,*Wert*-Paare in ein gewünschtes Format überträgt.

[101] `http://openbook.rheinwerk-verlag.de/javainsel9/javainsel_18_004.htm`, Zugriff 17.3.2019.

[102] Siehe hierzu Kapitel 5.3.2 und Abbildung 5.11.

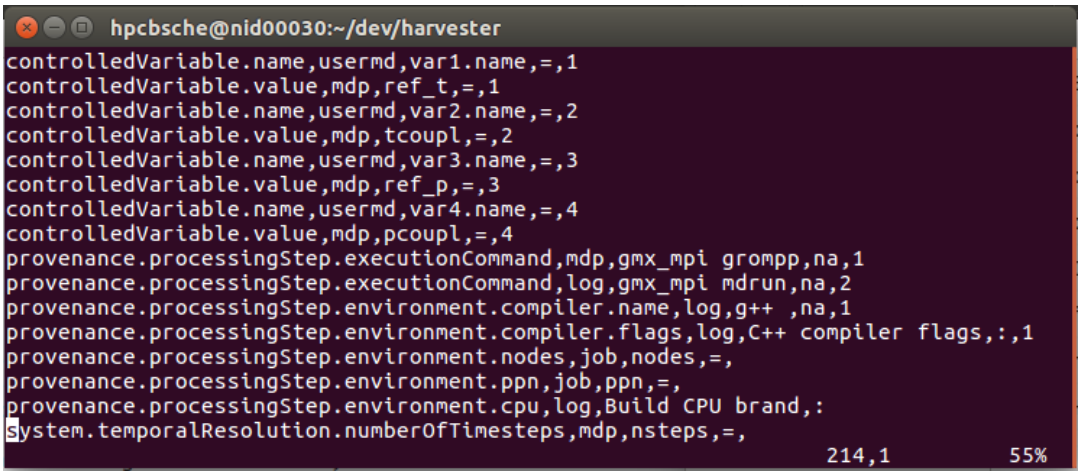

Abbildung 6.1: Ausschnitt einer Konfigurationsdatei zur Erfassung von GROMACS-Metadateninformationen.

Anpassbarkeit an Datenformate

Im Rahmen der Dissertation wurde die automatisierte Metadatenerfassung für drei Simulationscodes bzw. deren Datenformate getestet. Da die Anpassung der Metadatenerfassung an Simulationscodes bzw. Datenformate ausschließlich über die Definition einer Konfigurationsdatei für den jeweiligen Code erfolgt, wurden hierfür jeweils Konfigurationsdateien geschrieben. Diese nutzen die *EngMeta*-Metadatenschlüssel nach der in Kapitel 5.3.3 dargestellten Form statt der XSD-Definiton. Die Syntax der Konfigurationsdatei richtet sich dabei, wie in Kapitel 5.3.2 beschrieben, nach der Form $MDKey$,$Filename$,$SearchKey$, $Delimiter$, $Semantics$ und ist im Folgenden für die jeweiligen Simulationscodes im Appendix A angehängt. Es finden sich für die Simulationscodes jeweils die in der Evaluation automatisch aus den Ausgabedateien erfassten Metadaten mit ihrem Suchwort und ihrem Vorkommen sowie eine Besprechung.

GROMACS Für den Anwendungsfall des ITT, der GROMACS als Simulationscode einsetzt, wurde die Anpassbarkeit der Erfassung getestet. Die Ausgabedateien sind in Kapitel 1.6.1 erklärt. Ein Ausschnitt einer Konfiguration zur Erfassung von GROMACS-Metadateninformationen findet sich in Abbildung 6.1. Dort sind die Schlüssel gemäß dem *EngMeta*-Metadatenschlüssel definiert, gefolgt von der Datei, dem Suchbegriff, dem Delimiter und der Semantik, sofern benötigt. Die komplette Konfigurationsdatei, die auch für die Evaluation in Kapitel 6 verwendet wurde, findet sich in Anhang A.1. Die erfassten Metadaten als EngMeta-konformes XML finden sich in Anhang B.1. Eine Liste der potentiell automatisiert erfassbaren Informationen, bezogen auf den Simulationscode GROMACS, findet sich in Tabelle 6.2. Dabei sind insbesondere kontrollierte Variablen erfassbar. Der Name der jeweiligen kontrollierten Variablen muss aber in der benutzerdefinierten Datei mitgeteilt werden, was z.B. über die Zeile 1 wie in Listing 6.2 erfolgt. Die Einträge im Listing bedeuten beispielsweise, dass eine kontrollierte Variable existiert, deren Name in der Datei

Metadatenschlüssel (EngMeta entspr.)	Vorkommen	Suchwort/Zeile
processingStep.date	*.mdp	At date
controlledVariable.name	*.usermd	var1.name
controlledVariable.value	*.mdp	ref_t
controlledVariable.name	*.usermd	var2.name
controlledVariable.value	*.mdp	tcoupl
controlledVariable.name	*.usermd	var3.name
controlledVariable.value	*.mdp	ref_p
controlledVariable.name	*.usermd	var4.name
controlledVariable.value	*.mdp	pcoupl
processingStep.tool.name	*.log	GROMACS
processingStep.tool.softwareVersion	*.log	GROMACS version
processingStep.tool.operatingSystem	*.log	Build OS/arch
processingStep.executionCommand	*.log	gmx_mpi mdrun
processingStep.executionCommand	*.log	gmx_mpi grompp
processingStep.environment.compiler.name	*.log	C++ compiler
processingStep.environment.compiler.flags	*.log	C++ compiler flags
processingStep.environment.compiler.name	*.log	C compiler
processingStep.environment.compiler.flags	*.log	C compiler flags
processingStep.environment.nodes	*.job	nodes
processingStep.environment.ppn	*.job	ppn
processingStep.environment.cpu	*.log	Build CPU brand
system.grid.countX	*.gro	letzte Zeile
system.grid.countY	*.gro	letzte Zeile
system.grid.countZ	*.gro	letzte Zeile
system.temporalResolution.numberOfTimesteps	*.mdp	nsteps
system.temporalResolution.interval	*.mdp	dt

Tabelle 6.2: Auflistung der wesentlichen automatisiert erfassbaren Metadaten für GROMACS nach EngMeta-Spezifikation.

mit der Endung *usermd* unter dem Suchwort *var1.name* und nach dem Trenner = zu finden ist (Zeile 1). Der Wert der kontrollierten Variablen findet sich in der Ausgabedatei mit der Endung *mdp* unter dem Suchwort *ref_t* und dem Trenner = (Zeile 2). Die 1 am Ende jeder Zeile bedeutet, dass diese Informationen zu ein und derselben kontrollierten Variable gehören.

```
1 controlledVariable.name,usermd,var1.name,=,1
2 controlledVariable.value,mdp,ref_t,=,1
```

Listing 6.2: Angabe des Variablennamens und Assoziation mit dem geparsten Wert in der Konfigurationsdatei *fdm_itt.conf*.

Der Eintrag in der zugehörigen Datei *usermd* entspricht Listing 6.3. Hier werden die nötigen, nicht in den Ausgabedateien enthaltenen Informationen manuell eingefügt.

```
1 var1.name = temperature
2 var2.name = thermostat
```

Listing 6.3: Definition des Namens in der benutzerdefinierten Metadaten-Datei *run.usermd*.

Metadatenschlüssel (EngMeta entspr.)	Vorkommen	Suchwort/Zeile
description	ns3d.i	case_comment
controlledVariable.name	output.log	Mach
controlledVariable.value	output.log	Mach
controlledVariable.name	output.log	Reynolds
controlledVariable.value	output.log	Reynolds
controlledVariable.name	output.log	Prandtl
controlledVariable.value	output.log	Prandtl
controlledVariable.name	output.log	time
controlledVariable.value	output.log	time
processingStep.environment.compiler.name	output.log	compiled with
processingStep.environment.compiler.flags	output.log	18
processingStep.environment.nodes	*.job	nodes
processingStep.environment.ppn	*.job	ppn
processingStep.tool.name	*.log	Typ
processingStep.tool.softwareVersion	*.log	| Version
system.grid.countX	*.log	Grid points nx (Rank 0)
system.grid.countY	*.log	Grid points ny (Rank 0)
system.grid.countZ	*.log	Grid points nz (Rank 0)
system.grid.countCells	*.log	Total number of grid points
system.tmpResolution.numberOfTimesteps	ns3d.i	niter
system.tmpResolution.interval	ns3d.i	dt

Tabelle 6.3: Auflistung der wesentlichen automatisiert erfassbaren Metadaten von Flower/NS3D nach EngMeta-Konvention.

Was darüber hinaus komplett automatisiert erfasst werden kann, sind Informationen über die genutzte Software und das Betriebssystem, den Ausführungsbefehl, Informationen über den Compiler und die Rechnerumgebung sowie über die räumliche und zeitliche Auflösung der Simulation.

EAS3 Die Anpassung an das Datenformat EAS3 (genutzt vom Simulationscode NS3D) vom IAG erfolgt mittels der Analyse des Dateiformats[103]. Es zeigte sich, dass auch hier viele der Metadaten schon in den Ein- oder Ausgabedateien der Simulationscodes verfügbar sind. Eine Auflistung findet sich in Tabelle 6.3. Dabei lassen sich insbesondere die Beschreibung, kontrollierten Variablen, Compiler und Tool-Informationen sowie Informationen zur zeitlichen und räumlichen Auflösung der Simulation automatisiert auslesen. Die komplette Auflistung bzw. die Konfigurationsdatei findet sich in Anhang A.2. Das Ergebnis der Erfassung als EngMeta-konformes XML ist in Appendix B.2 dargestellt.

NetCDF Um die Erweiterbarkeit auf andere, nicht in den ursprünglichen Anwendungsfällen mitbetrachteten Simulationscodes und Datenformate zu testen, wurde die Übertragung auf NetCDF evaluiert. NetCDF (Network Common Data Format)[104]

[103] Beschrieben in Kapitel 1.6.2.

[104] `https://www.unidata.ucar.edu/software/netcdf/`, Zugriff 16.3.2019.

ist ein speziell in den Klimawissenschaften und in der Erdsystemmodellierung verbreitetes Datenformat, welches Metadaten in einem Container enthält. Dabei liegen Bibliotheken für alle im Höchstleistungsrechnen relevanten Programmiersprachen vor, die dieses Datenformat schreiben können. Die Daten liegen nach der Simulation im Binärformat mit der Dateiendung *.nc* bereit, die Metadaten können mit Hilfe des Programms *ncdump* in eine gesonderte Datei mit der Endung *.cdl* geschrieben werden. Für die Metadatendateien gibt es verschiedene Standards[105].

```
1  netcdf sresa1b_ncar_ccsm3-example {
2  dimensions:
3    lat = 128 ;
4    lon = 256 ;
5    bnds = 2 ;
6    plev = 17 ;
7    time = UNLIMITED ; // (1 currently)
8  variables:
9    float area(lat, lon) ;
10     area:long_name = "Surface area" ;
11     area:units = "meter2" ;
12 [..]
```

Listing 6.4: Beginn einer NetCDF-Metadatendatei *sresa1b_ncar_ccsm3-example.cdl* der UCAR-Beispieldatensätze.

Listing 6.4 stellt den Beginn der Metadatendatei zum Datensatz *sresa1b_ncar_ccsm3-example* nach *Climate and Forecast* (CF) Konvention[106] dar, welche eine Festlegung für die Klimawissenschaften verkörpert. Der zugehörige Datensatz *sresa1b _ncar_ccsm3-example.nc* stammt aus dem *Community Climate System Model* (CCSM) und repräsentiert einen simulierten Zeitschritt von Niederschlag, Lufttemperatur und Wind in Richtung Osten. In der Metadatendatei zeigen sich die Informationen über die Variablen und ihre Namen, Wertebereiche, Einheiten usw. Beispielsweise findet sich ab Zeile 9 die Variable *area* definiert (*float area(lat, lon)*), die durch einen Namen (*area:long_name = „Surface area“ ;*) und die Einheit (*area:units = „meter2“ ;*) charakterisiert wird.

Dieser Datensatz wurde auf der Cray URIKA nach Metadaten automatisiert ausgewertet, wozu eine Konfigurationsdatei zur Erfassung der Metadaten geschrieben werden musste. Diese findet sich in Appendix A.3. In Tabelle 6.4 sind die erfassten Metadatenkeys mit ihrem jeweiligen Ort und Suchbegriff dargestellt. Hier findet sich auch die in Listing 6.4 ab Zeile 9 definierte Variable wieder, die in der Tabelle durch ein hochgestelltes * angezeigt ist. Die erfassten Metadaten gemäß EngMeta-Standard sind in Anhang B.3 als XML dargestellt.

[105] `https://www.unidata.ucar.edu/software/netcdf/examples/files.html`, Zugriff 16.3.2019.

[106] `http://cfconventions.org/`, Zugriff 13.4.2019.

Metadatenschlüssel (EngMeta entspr.)	Vorkommen	Suchwort/Zeile
contact.affiliation.name	cdl	institution
contact.email	cdl	contact
project.value	cdl	project_id
title	cdl	title
controlledVariable.name*	cdl	float area
controlledVariable.symbol*	cdl	area:long_name
controlledVariable.encoding*	cdl	area:units
controlledVariable.name	cdl	tas:standard_name
controlledVariable.value	cdl	tas:_FillValue
controlledVariable.symbol	cdl	float tas
controlledVariable.encoding	cdl	tas:unit
controlledVariable ...	cdl	...
processingStep.type	cdl	experiment_id
processingStep.method.description	cdl	comment
processingStep.input.id	cdl	ozone forcing
processingStep.input.id	cdl	aerosol optics
processingStep.input. ...	cdl	...
processingStep.tool.name	cdl	source
processingStep.tool.referencedPublication.citation	cdl	references
processingStep.executionCommand	cdl	cmd_ln
rightsStatement.copyrightInformation.note	cdl	acknowledgment

Tabelle 6.4: Auflistung der wesentlichen automatisiert erfassbaren Metadaten von CCSM/NetCDF nach EngMeta-Konvention.

Über alle hier evaluierten Simulationscodes hinweg lässt sich feststellen, dass sich aus den Ausgabedaten der Simulationscodes – wie in Kapitel 5.3 diskutiert – insbesondere fachspezifische und prozessspezifische Metadaten automatisiert erfassen lassen. Dabei können in allen evaluierten Fällen die kontrollierten Variablen ausgelesen werden, da Informationen hierzu stets mit ausgegeben werden. Ebenso können weitere fachspezifische Informationen über das simulierte Zielsystem bei EAS3 und GROMACS erfasst werden, nämlich die räumlichen und zeitlichen Informationen. Bei allen drei getesteten Datenformaten können viele prozessspezifische Informationen ausgelesen werden, insbesondere Informationen zu den Compilern, Ausführungsbefehlen, Softwareversionen und zur genutzten Rechenumgebung. Da das NetCDF-Containerformat nach der CF-Konvention auch Beschreibungsmöglichkeiten enthält, konnten hier auch einige deskriptive Metadaten erfasst werden, so z.B. die Kontaktdaten sowie ein beschreibender Titel.

Grundsätzlich lässt sich feststellen, dass sich weniger Metadateninformationen erfassen lassen als erhofft. Es hängt immer von der Ausgabe der Simulationscodes ab, wie viele Informationen sich wie gut extrahieren lassen. Eine höhere Standardisierung in der Ausgabe der Simulationscodes erleichtert dabei das Erstellen der Konfigurationsdatei und somit das Parsing. Beispielsweise richtet sich die Metadateninformation, die NetCDF in den *cdl*-Dateien ausgibt, nach der CF-Konvention und ist somit für alle Simulationscodes, die NetCDF mit der CF-Konvention nutzen, gleich. Dies macht es

einfach, die Metadatenerfassung für den Code anzupassen, wenn die Konvention und die Semantik jedes Feldes klar ist. Je stärker standardisiert die schon vorhandene Struktur, desto einfacher lassen sich die Konfigurationsdateien schreiben und lässt sich die Erfassung durchführen. GROMACS beispielsweise verwendet in allen für die Erfassung relevanten Dateien *.gro*, *.mdp* und *.log* unterschiedliche Syntax für unterschiedliche Metadaten. So ist z.B. die Geometrieinformation in der letzten Zeile der *.gro*-Datei codiert, die *.log*-Dateien nutzen als Trenner einen Doppelpunkt, die *.mdp*-Dateien dagegen ein Gleichheitszeichen. Diese unterschiedliche Syntax erschwert das Erstellen von Konfigurationsdateien. Bei NetCDF dagegen wird eine Syntax stringent durch eine einzige *.cdl*-Metadatendatei eingehalten, was das Erstellen von Konfigurationsdateien erleichtert.

Abgleich mit den Anforderungen Bezogen auf die Anforderungen, die in Kapitel 4 aufgestellt wurden, lässt sich Folgendes feststellen: Die funktionale Anforderung FA 1.7 (Metadaten müssen automatisch extrahiert werden, sofern möglich) ist erfüllt, da ein Werkzeug vorliegt, mit dem sich insbesondere die technischen sowie die Prozess- und die fachspezifischen Metadaten erfassen lassen. Dies wurde für die Datenformate bzw. Simulationscodes GROMACS, EAS3 und NetCDF gezeigt und stellt somit einen generischen, übertragbaren Ansatz dar. Ebenso können individuelle Readme-Dateien von der Erfassung ausgelesen werden, sofern sie einer textuellen Formalisierung entsprechen. Der Grad der automatisierten Erfassbarkeit von Metadaten ist immer simulationscode- bzw. datenformatabhängig. Es kommt immer darauf an, wie viele Ausgaben der Simulationscode automatisch mitliefert. Insbesondere bei selbst entwickelten Codes steckt hier noch viel Spielraum zur Erhöhung der Metadatenausgabe, die dann einfach von der Erfassung weiterverarbeitet werden kann. Allerdings ist das Schreiben der Konfigurationsdatei immer mit Aufwand verbunden, insbesondere wenn, wie bei GROMACS, die verschiedenen Ausgabedaten jeweils eine verschiedene Syntax haben. Möglich ist die Erfassung jedoch in jedem Fall, wenn die Werte in einer beliebigen Form als (*Suchschlüssel, Trenner, Wert*)-Tripel vorliegen.

Rechnerumgebungen

Im Folgenden soll die Adaptierbarkeit der automatisierten Metadatenerfassung für verschiedene, im Arbeitsbereich typische Rechnerumgebungen evaluiert werden. Typische Rechnerumgebungen sind hierbei, neben dem Arbeitsplatzrechner selbst, die verschiedenen Cluster- und Höchstleistungsrechnersysteme, die auch von den Wissenschaftlerinnen und Wissenschaftlern zur Simulation genutzt werden.

Arbeitsplatzlaptop Auf einem Arbeitsplatzrechner unter Linux mit Ubuntu 18.04 lässt sich die automatisierte Metadatenerfassung ohne Probleme starten und auf ein Verzeichnis anwenden, in dem dann Metadaten erfasst werden. Die native Variante funktioniert immer, sofern Java 1.8 verfügbar ist. Die parallele Variante kann auch auf einem Arbeitsplatzlaptop benutzt werden, wenn das Spark Analyseframework im

Standalone-Modus betrieben wird[107]. Die automatisierte Metadatenerfassung lässt sich auch auf Windows 10 zur Ausführung bringen, wobei die parallele Erfassung mangels offizieller Kompatibilität von Spark mit Windows nicht möglich ist bzw. nicht getestet werden konnte. Für die Ausführung der nativen Variante der Erfassung waren keinerlei Anpassungsarbeiten nötig, da das Java-Programm durch den Bytecode, wie erwartet, plattformunabhängig funktioniert.

bwUniCluster Der *bwUniCluster*[108] ist ein Clustersystem auf Leistungsebene 3 und stellt Ressourcen für Studierende, Doktorandinnen und Doktoranden sowie allgemeine wissenschaftliche Rechenprojekte landesweit zur Verfügung[109]. Das System wird mit Red Hat Enterprise Linux (RHEL) 7.5 betrieben. Da das ITT zum großen Teil auf diesem System rechnet, wurde auch dieses System getestet. Bei den Tests zeigte sich, dass die native Variante ohne Probleme ausgeführt werden kann. Zur Nutzung der parallelen Variante musste zunächst Spark manuell lokal installiert werden[110].

Cray XC40 Hazel Hen Die Cray XC40 „Hazel Hen“ ist ein Höchstleistungsrechner der obersten Leistungsebene (Tier 0) und stellt Rechenressourcen bundes- und europaweit für wissenschaftliche Rechnungen aller Art zur Verfügung[111]. Da es das Hauptsystem am HLRS darstellt und auch das IAG als Anwendungsfall hauptsächlich dieses System simuliert, wurde die Erfassung auf diesem System getestet. Es wird mit dem Cray Linux Environment (CLE) Version 6.0.UP05 betrieben, welches seinerseits auf SuSe Linux Enterprise Edition Version 12SP3 basiert. Zunächst stürzte die Erfassung mit dem Fehler *UnsupportedClassVersionError* ab. Es stellte sich heraus, dass standardmäßig statt des von der Erfassung benötigten Java 1.8 nur die Version 1.7. geladen war. Bei Nutzung dieses Systems muss also mittels des Befehls in Listing 6.5 Java 1.8 nachgeladen werden.

```
1 hpcbsche@eslogin002:~> module load java
```

Listing 6.5: Java 1.8 auf der Cray XC40 nachladen

Auf der Cray XC40 konnte nur die native, auf der Scanner API basierende Variante zur Ausführung gebracht werden. Die parallele Variante wurde nicht zur Ausführung gebracht, da die Sicherheitsrichtlinien des HLRS die Ausführung des notwendigen Datenanalyse-Frameworks Spark verbieten.

[107] http://spark.apache.org/docs/latest/spark-standalone.html, Zugriff 12.4.2019.
[108] https://www.bwhpc-c5.de/wiki/index.php/Category:BwUniCluster, Zugriff 15.3.2019
[109] Entsprechend der Klassifikation der Leistungsebenen nach https://mwk.baden-wuerttemberg.de/fileadmin/redaktion/m-mwk/intern/dateien/pdf/Forschung/Umsetzungskonzept_bwHPC.pdf, Zugriff 17.4.2019.
[110] https://www.bwhpc-c5.de/wiki/index.php/Spark, Zugriff 15.3.2019.
[111] https://www.hlrs.de/systems/cray-xc40-hazel-hen/, Zugriff 15.3.2019.

	nativ Scanner API	parallel Apache Spark
Arbeitsplatzrechner		
Ubuntu 18.04	✓	✓
Windows 10	✓	–
bwUniCluster		
RHEL 7.5	✓	✓
Cray XC40		
CLE 6.0.UP05/SLES 12SP3	✓	–
Cray URIKA		
Urika-GX-2.2UP00/CentOS 7.4	✓	✓

Tabelle 6.5: Überblick über die unterstützten Rechnerplattformen.

Cray URIKA Am HLRS ist die Cray URIKA die Hauptplattform für „Big Data Analytics“[112]. Der getestete Betriebssystem-Stack *Urika-GX-2.2UP00* basiert auf *CentOS 7.4.1708* und enthält die CrayGraph Engine, Spark und Hadoop als mögliche Analyse-Frameworks. Auf dieser Plattform ließen sich sowohl die Scanner- als auch die Spark-Variante der Erfassung problemlos ausführen.

Abgleich mit den Anforderungen Tabelle 6.5 zeigt die Zusammenfassung der Evaluationsergebnisse hinsichtlich der verschiedenen Rechnerumgebungen. Dabei ist ersichtlich, dass die native Variante auf einer Vielzahl von Betriebssystemen ausführbar ist. Dabei wurden sowohl Red Hat als auch verwandte Systeme, teilweise in ihren jeweiligen Großrechnerausprägungen getestet und mit Ubuntu ebenso ein Vertreter der Debian-Linie. Darüber hinaus wurde die Erfassung auch für das Windows-Betriebssystem getestet. Dieser Test scheint zunächst unwichtig zu sein, da in der gegenwärtigen Situation Metadaten von Daten erfasst werden sollen, die fast ausschließlich auf Linux- oder unixoiden Betriebssystemen produziert wurden. Im Hinblick auf die Erweiterbarkeit spielt die Erfassung auf Windows-Plattformen aber sehr wohl eine Rolle: Die Auswertung von Experimentaldaten erfolgt teilweise mittels Rechnern basierend auf dem Windows-Betriebssystem, sodass dieser Test von Bedeutung für die Portabilität der Lösung war. Die parallele Variante funktioniert nur dort, wo das Datenanalyse-Framework Spark verfügbar war. Bei keinem der Tests waren Administratorprivilegien nötig, und die Ausführung erfolgte lokal im Workspace. Zusatzsoftware musste für die native Variante nicht installiert werden, auf der Cray XC40 musste lediglich die korrekte Version 1.8 von Java nachgeladen werden. Damit sind die nicht-funktionalen Anforderungen NFA 1.1 (automatisierte Metadatenextraktion muss ohne Administratorprivilegien lauffähig sein), NFA 1.2 (automatisierte Metadatenextraktion muss so wenig Zusatzsoftware wie möglich benötigen) sowie NFA1.3 (Metadatenextraktion muss lokal in einem HPC Workspace stattfinden) erfüllt.

[112] `https://www.hlrs.de/systems/cray-urika-gx/`, Zugriff 15.3.2019.

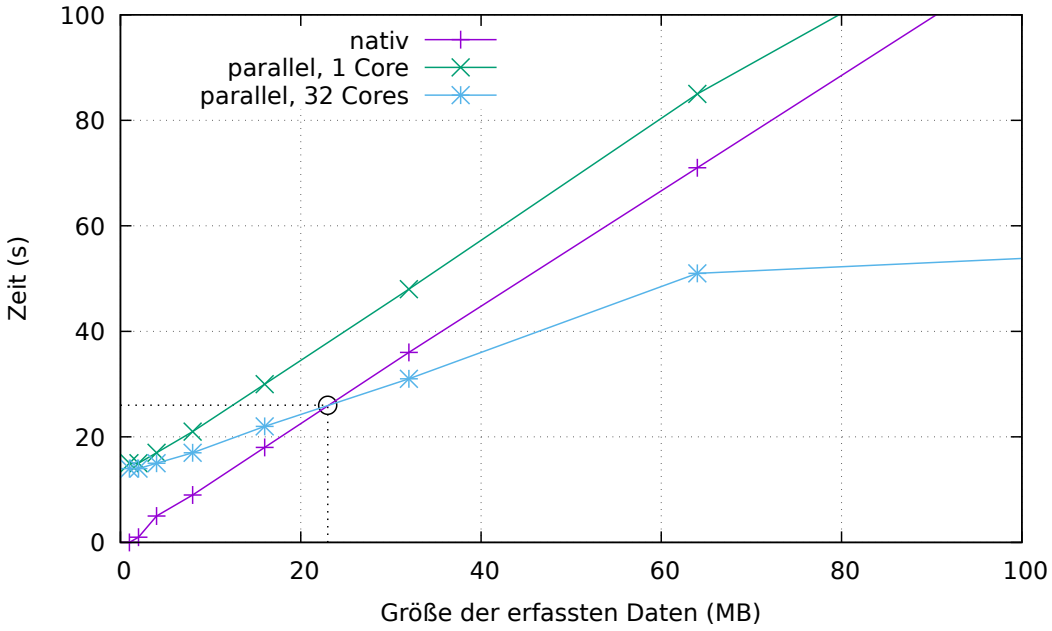

Abbildung 6.2: Vergleich der Ausführungsgeschwindigkeiten der nativen (Java Scanner) Variante und der parallelen (Java + Apache Spark) Variante. Verglichen auf der Cray URIKA.

6.3.2 Performance

Die Performance der automatisierten Metadatenerfassung ist nur insofern von Bedeutung, als sie den Arbeitsprozess nicht zu lange unterbrechen sollte. Die Lösung wird hinsichtlich der Anforderung, dass die Ausführungszeit der Metadatenerfassung im Verhältnis zur Ausführungszeit der Simulation sein muss, evaluiert. Die Perfomancetests wurden dabei auf der Cray URIKA am HLRS durchgeführt. Dieses System besteht aus 41 Rechenknoten, wovon jeder über folgende Eigenschaften verfügt:

- Prozessor: 2x Intel „Broadwell“ E5-2695V4@2,1 GHz (2x 18 Kerne)
- Hauptspeicher: 512 GB
- Festspeicher: Anbindung an 240 TB Lustre (Sonexion 900) mit 4.0 GB/s
- Software-Stack: *Urika-GX-2.2UP00* basierend auf *CentOS 7.4.1708*

Für weitere Tests wurde der bwUniCluster herangezogen. Aus Gründen der Vergleichbarkeit wurden dort die Knoten mit Broadwell-Prozessortechnologie verwendet. Sie verfügen über folgende Konfiguration:

- Prozessor: 2x Intel „Broadwell“ E5-2660V4@2,0 GHz (2x 14 Kerne)
- Hauptspeicher: 128 GB
- Festspeicher: Anbindung an 853 TiB Lustre mit 16.0 GB/s
- Software-Stack: RHEL 7.5

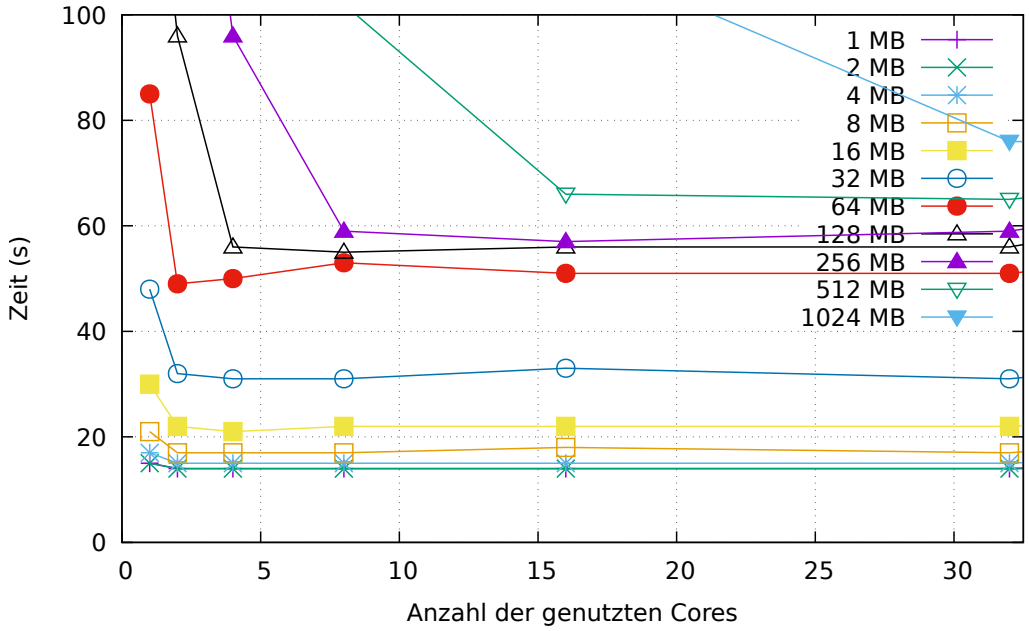

Abbildung 6.3: Skalierungstests für die parallele Variante auf der Cray URIKA. Ausführungszeit im Verhältnis zur Anzahl der benutzten Kerne für die Parallelisierung.

Die Testdatensätze, die aus GROMACS-Logdateien bestehen, wurden auf dem Lustre abgelegt und von dort erfasst. Die native Variante wurde auf dem Login-Knoten, welcher die gleiche Ausstattung wie die Rechenknoten hat, gestartet. Die parallele wurde mittels des Schedulers *Mesos* auf der Cray URIKA auf einem oder mehreren Rechenknoten zur Ausführung gebracht, auf dem bwUniCluster wurden manuell Knoten reserviert und das Spark-Framework im Cluster-Modus gestartet.

Vergleich native und parallele Variante Abbildung 6.2 zeigt die Gegenüberstellung der nativen (mittels Scanner API) und der parallelen (mittels Apache Spark) Lösung. Die native Variante verhält sich linear mit der Größe der auszuwertenden Daten, wie in Kapitel 5.3.2 auch theoretisch bestimmt wurde. Die parallele Variante verhält sich auf einem Core ausgeführt ebenso linear und liegt durch den Overhead, den das Spark Datenanalyseframework erzeugt, noch 15 Sekunden über der Ausführungszeit der nativen Implementierung. Die Ausführungszeit der parallelen Implementierung, auf 32 Kernen ausgeführt, wächst schwach linear mit der Eingabegröße, vermutlich $\mathcal{O}(\sqrt{n})$. Es zeigt sich, dass die native Lösung bis zu einer Größe der erfassten Daten von 23 MB schneller ist als die parallele Variante. Erst ab dieser Datengröße kann von der parallelen Lösung bei der Metadatenerfassung profitiert werden. Dies bedeutet insbesondere, dass die native Lösung für die meisten Fälle ausreichend sein wird, da die Größe der Logdateien von Simulationscodes in der Regel im Bereich von $\mathcal{O}(1MB)$ - $\mathcal{O}(10MB)$ liegt. GROMACS erzeugt aber auch Logdateien von 30 oder mehr Megabytes Größe.

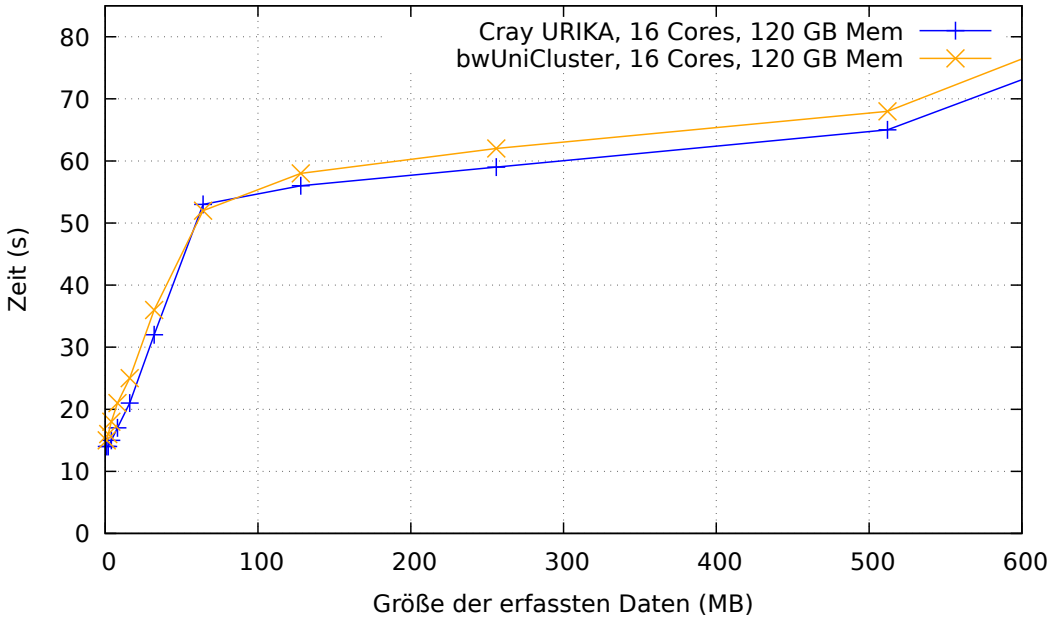

Abbildung 6.4: Vergleich der Ausführungsgeschwindigkeit der parallelen Variante auf der Cray URIKA und auf dem bwUniCluster.

Skalierung parallele Variante auf Cray URIKA Auf der Cray URIKA wurden Skalierungstests für die parallele Variante der Metadatenerfassung durchgeführt. Dazu wurde die Ausführungszeit bei wechselnder Anzahl von Kernen für verschiedene Datengrößen gemessen. Dieser Zusammenhang ist in Schaubild 6.3 dargestellt. Es zeigt sich, dass der positive Effekt der Parallelität für Daten unter 256 MB schon bei einer Anzahl von 8 Kernen eintritt und sich nicht mehr wesentlich verbessert. Nur große Datensätze von über 512 MB profitieren von noch mehr Kernen und können mit 16 oder 32 Kernen ihre Laufzeit deutlich verbessern[113].

Vergleich parallele Variante auf Cray URIKA und bwUniCluster Die automatisierte Metadatenerfassung mittels der parallelen Variante wurde schließlich auf zwei Rechnerplattformen getestet, die das Spark-Datenanalyseframework bereitstellen. Neben der Cray URIKA wurde die Ausführungsgeschwindigkeit auf dem bwUniCluster getestet. Da dieser nur über 128 GB Hauptspeicher pro Knoten verfügt, wurde für beide Maschinen die Hauptspeichernutzung der Erfassung auf 120 GB limitiert. Die Ausführung wurde auf beide Maschinen mit 16 Kernen getestet. Abbildung 6.4 zeigt die Ausführungsgeschwindigkeiten auf beiden Maschinen für verschiedene Datengrößen im Vergleich. Dabei ist zu erkennen, dass die Cray URIKA für alle getesteten Werte, bis auf einen[114], um wenige Sekunden schneller war als der bwUniCluster.

[113] In weiteren Tests zeigte sich, dass mit 40 Kernen alle Laufzeiten zunächst wieder angestiegen sind, was in obiger Grafik nicht mehr zu sehen ist. Dieses Verhalten rührt von der Tatsache her, dass ein Knoten auf der Cray URIKA über 36 Kerne verfügt. Ab einer höheren Anzahl von Kernen wird ein weiterer Knoten hinzugenommen, was wiederum zu erhöhter Kommunikation und damit zu schlechterer Performance führt.

[114] Bei 64 MB zeigt sich der bwUniCluster um 1,4 s schneller als die Cray URIKA.

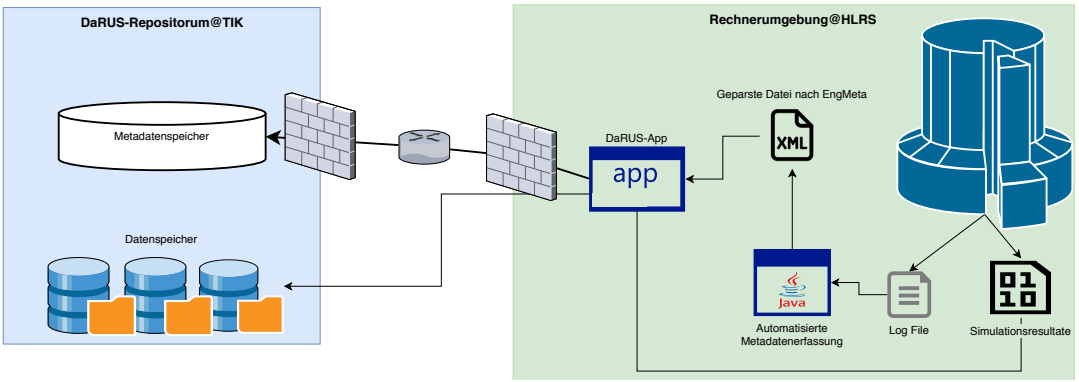

Abbildung 6.5: Netzwerkbereiche und Arbeitsprozess von der Metadatenerfassung bis zum Einspielen in ein Repositorium.

Abgleich mit den Anforderungen Zusammenfassend lässt sich feststellen, dass Anforderung NFA 1.4 erfüllt wird, da die automatisierte Metadatenerfassung für Logdateigrößen von bis zu 50 MB weniger als 60 Sekunden Ausführungszeit benötigt. Typische Größen bewegen sich im Bereich von wenigen Megabytes, wofür die native Variante nur wenige Sekunden benötigt.

6.4 Integration in den Arbeitsprozess

In diesem Unterkapitel wird die Integration in den Arbeitsprozess gezeigt. Dies umfasst insbesondere die Ermöglichung des Datenflusses, wie er in Abbildung 6.5 dargestellt ist. Die in der Simulation erzeugten Logdateien müssen durch die automatisierte Metadatenerfassung ausgewertet und in eine XML-Datei umgewandelt werden. Die DaRUS-App nimmt dieses XML als Eingabe und erstellt einen Datensatz mit Metadatenbeschreibung (gemäß *EngMeta*) im DaRUS-Repositorium des TIK.

6.4.1 Integration in die Großrechnerumgebung

Die Metadatenerfassung kann, wie in Kapitel 5.3.4 beschrieben, manuell durch den Aufruf des Wrapper-Skripts durchgeführt werden, wann immer sie benötigt wird. Allerdings kann die Erfassung auch direkt im Job-Skript für den Scheduler definiert werden, um direkt nach dem Simulationslauf die Metadatenerfassung zu starten. Dabei wird sie durch den Scheduler auf einem der MOM-Knoten zur Ausführung gebracht. Ein Beispiel, hier bezogen auf die Cray XC40 Rechnerumgebung mit dem PBS-Scheduler, findet sich in Listing 6.6. Dabei muss zunächst Java 1.8 geladen werden (Zeile 6). Dem Aufruf des parallelen Simulationscodes (Zeile 12) wird einfach der Aufruf der automatisierten Metadatenerfassung nachgestellt (Zeile 13). Dort wird das eigentliche Erfassungsprogramm $fdm.sh$ mit den entsprechenden Parametern aufgerufen. Der erste Parameter gibt die Konfigurationsdatei an, der zweite referenziert auf das aktuelle Ausführungsverzeichnis der Simulation. Der letzte Parameter mit dem Wert $scanner$ führt die Erfassung mit der nativen Variante aus.

```
1  #!/bin/bash
2  #PBS -N Aero_Simulation
3  #PBS -l nodes=1:ppn=24
4  #PBS -l walltime=00:20:00
5  #PBS -M schembera@hlrs.de
6  module load java
7
8  # Change to the direcotry that the job was submitted from
9  cd $PBS_O_WORKDIR
10
11 # Launch the parallel job and the metadata collection right after
12 aprun -n 24 -N 24 ~/promotion/aeroCode > my_output_file 2>&1
13 ~/harvester/fdm.sh ~/harvester/fdm_iag_eval.conf . scanner
```

Listing 6.6: Ausführung eines parallelen Simulationscodes (Zeile 12) mit anschließender automatisierter Metadatenerfassung (Zeile 13) in einer Job-Datei.

Prinzipiell lässt sich die Erfassung auch als Epilog-Skript realisieren. Dies könnte durch eine Direktive in der Job-Datei wie in Listing 6.7 abgebildet geschehen. Allerdings werden Epilog-Skripte unter *root*-Privilegien gestartet, die für die Erfassung eigentlich nicht nötig sind. Epilog-Skripte eignen sich eher für zentrale Nach- oder Aufräumarbeiten, die mehrere Rechenknoten betreffen. Allerdings könnte sich damit eine zentrale Metadatenerfassung für eine ganze Institution einheitlich implementieren lassen, sofern dieser Anwendungsfall vorkommt.

```
1  #PBS -l epilogue=~/harvester/fdm.sh ~/harvester/fdm_iag_eval.conf .
       scanner
```

Listing 6.7: Automatisierte Metadatenerfassung als Epilog-Skript.

Abgleich mit den Anforderungen Indem der automatisierte Erfassungsvorgang nach einer Simulation gezeigt wurde, kann die Anforderung FA 2.2 (die Nutzerschnittstelle muss Metadatenannotation ermöglichen) und FA 2.5 (die Metadatenerfassung muss automatisiert nach der Simulation ausführbar sein) erfüllt werden. Da die Erfassungssoftware allein mit Nutzerprivilegien lauffähig ist, kann auch die nichtfunktionale Anforderung NFA 1.1 (die automatisierte Metadatenextraktion muss ohne Administratorprivilegien lauffähig sein) als erfüllt gelten, ebenso die NFA 1.3 (die Metadatenextraktion muss lokal in einem HPC Workspace stattfinden können). Die Erfüllung von NFA 1.2 (die automatisierte Metadatenextraktion muss so wenig Zusatzsoftware wie möglich benötigen) wurde bereits in Kapitel 6.3.1 gezeigt.

6.4.2 Integration mit einem Repositorium

Schließlich können die Daten mit den erfassten Metadaten direkt in das DaRUS-Repositorium gespielt werden. Nachdem die Erfassung durchgeführt wurde, stehen die

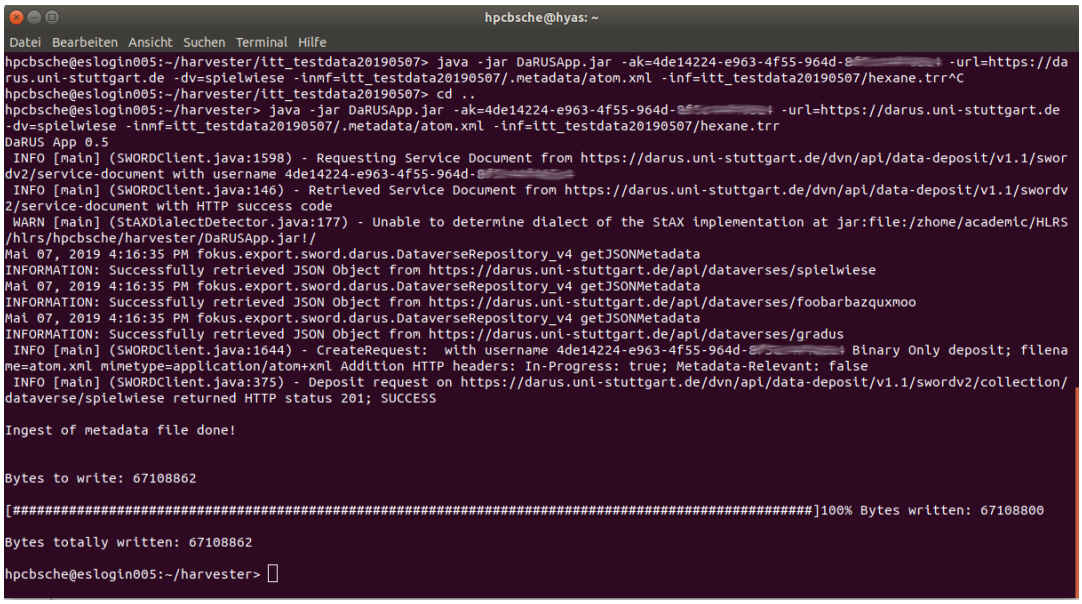

Abbildung 6.6: Screenshot des Einspielvorgangs von Daten und erfassten Metadaten mittels der DaRUS-App in das Dataverse-Repositorium vom Frontend-Knoten der Cray XC40.

Metadaten im Unterverzeichnis *.metadata* bereit. Beim Aufruf der DaRUS-App (siehe Kapitel 5.1), welche die Schnittstelle in das Dataverse-Repositorium darstellt, wird die Metadatendatei angegeben sowie der Name der Datei, die übertragen werden soll. Darüber hinaus muss nach dem Sicherheitsschlüssel die URL des Repositoriums sowie das logische Dataverse angegeben werden. Der Beispielaufruf der DaRUS-App findet sich in Listing 6.8. Im Mai 2019 ist nur das Einspielen von deskriptiven Metadaten im Rahmen der Sword-API möglich[115], weswegen hier die *atom.xml*-Datei als Metadatendatei angegeben ist. In Abbildung 6.6 findet sich der Einspielvorgang als Screenshot aus der Sicht der simulierenden Wissenschaftlerinnen und Wissenschaftler, die vom Frontend-Knoten des Großrechners XC40 einen Datensatz und die zugehörigen Metadaten mit der DaRUS-App in das Repositorium spielen. In Abbildung 6.7 ist die Weboberfläche des Repositoriums und der eingespielte Datensatz mit Metadaten, der noch unveröffentlicht ist, dargestellt.

```
1 java -jar DaRUSApp.jar -ak=4de14224-e461-4252-964d-xxxxxxxxxx -url=
    https://darus.uni-stuttgart.de -dv=spielwiese -inmf=.metadata/
    atom.xml -inf=hexane.trr
```

Listing 6.8: Befehl zum Einspielen von Daten in Dataverse mittels *DaRUS App*.

Die Vorbedingung zur Integration mit einem Repositorium ist die Erreichbarkeit des Repositoriums von der Plattform, auf dem die Erfassung und schließlich die DaRUS-App ausgeführt wurde. Hierzu war eine Anpassung der Firewallregeln sowohl auf Seiten des DaRUS-Dienstes als auch auf Seiten der Großrechnerumgebung nötig (siehe Firewalls in Abbildung 6.5). Dies wurde für die Frontend-Knoten eingerichtet.

[115] `http://guides.dataverse.org/en/latest/api/sword.html`, Zugriff 13.2.2019

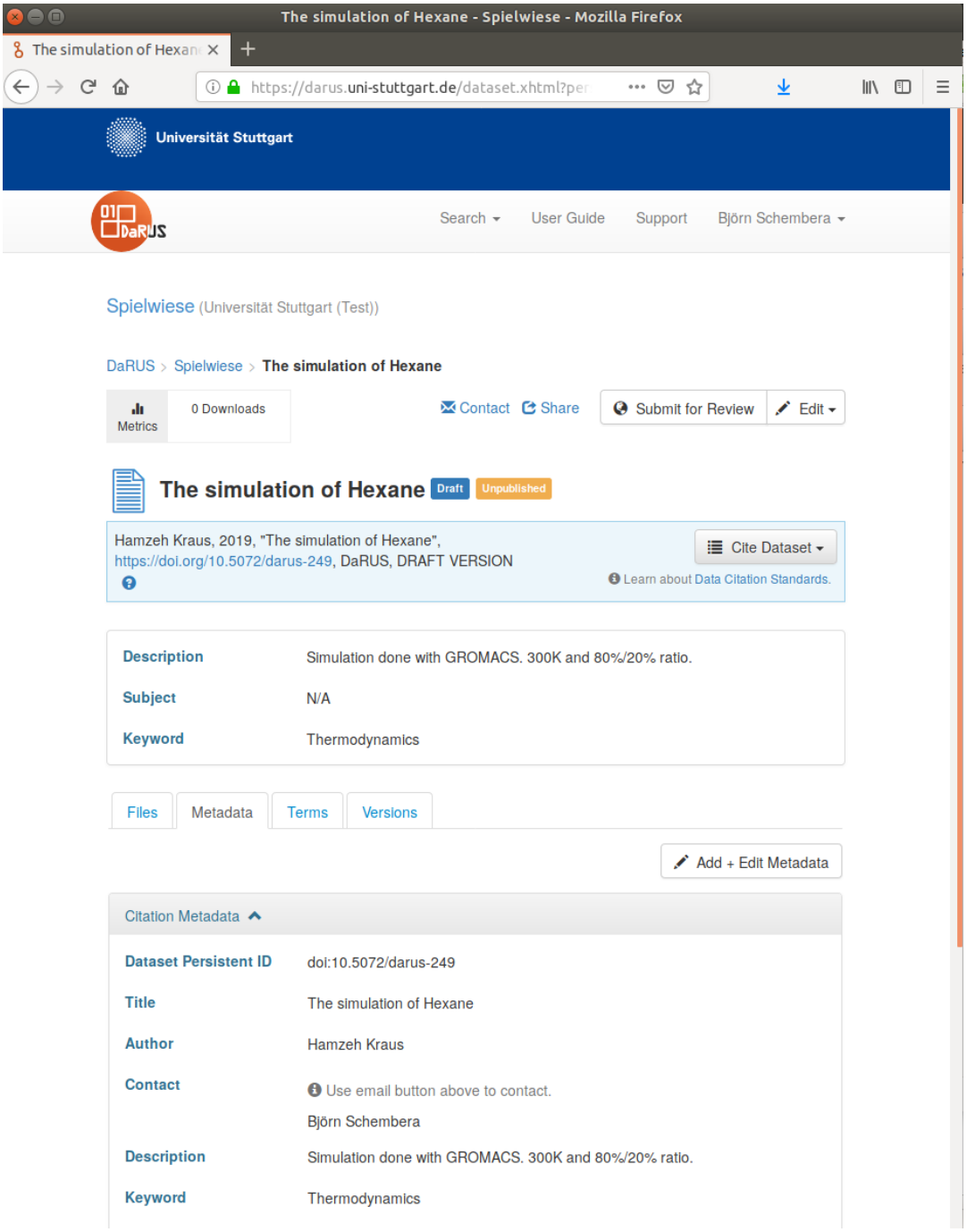

Abbildung 6.7: Der automatisiert eingespielte Datensatz auf dem DaRUS-Repositorium. Dieser ist noch unveröffentlicht.

Der Aufruf der DaRUS App wie in Listing 6.8 ließe sich auch an das Job-Skript anhängen, indem obiger Befehl einfach ans Ende von Listing 6.6 gesetzt wird.

Für diese vollautomatisierte Variante müssten auf einigen Systemen für alle internen Knoten die Firewallregeln geändert werden, da hier die DaRUS-App auf den MOM-Knoten läuft. Dies lässt sich umgeben, indem das Job-Skript in ein weiteres Skript gehüllt wird. Aus diesem Skript wird dann der *qsub*-Befehl aufgerufen und direkt danach die Befehle zur Metadatenerfassung und zum Einspielen in das Repositorium. Durch diese Vorgehensweise laufen die beiden Java-Programme auf dem Login-Knoten des jeweiligen Computersystems. Allerdings werden in der Praxis die Nutzerinnen und Nutzer der Lösung das Hochladen der Daten eher manuell vollziehen, um die Kontrolle darüber zu behalten, was genau in das Repositorium gespielt wird.

Abgleich mit den Anforderungen Damit können die Anfordungen FA 2.2 (die Nutzerschnittstelle muss Metadatenannotation ermöglichen) und FA 2.3 (die Schnittstelle muss als Kommandozeilentool verfügbar sein) als erfüllt gelten. Da die DaRUS-App auch als grafische Benutzeroberfläche zur Verfügung steht und das Repositorium über eine Weboberfläche zu bedienen ist, kann auch die FA 2.4 (die Schnittstelle muss als grafische Oberfläche verfügbar sein) als erfüllt gelten. Da sich die Kette *Metadatenerfassung – Einspielen der Daten+Metadaten in ein Repositorium* vollständig automatisieren lässt, wird die nicht-funktionale Anforderung NFA 2.2 (die Nutzerschnittstelle muss sich einfach in den HPC-Workflow integrieren lassen) erfüllt. Ebenso müssen die Nutzerinnen und Nutzer so – abgesehen von der DaRUS-App – keine Zusatzsoftware installieren. Das System ist auf Java 1.8. lauffähig, welches auf allen getesteten Rechnerumgebungen verfügbar war.

6.5 Scientific Data Officer

Die Rolle des SDO ist zum gegenwärtigen Stand der Arbeit nicht zu evaluieren. Allerdings werden einige Evaluationsansätze diskutiert. Nach Einführung der Rolle des SDO müssten zunächst die Bestände erhoben werden, z.B. bzgl. dunkler Daten oder dem generellen Zustand des Forschungsdatenmanagements in der Organisationseinheit. Schließlich müsste zu festgelegten Zeitpunkten periodisch überprüft werden, ob sich der Bestand der dunklen Daten verringert hat und der Prozess des Forschungsdatenmanagements weniger Hemmschwellen darstellt, da es die Aufgabe des SDO sein sollte, diese abzubauen. Ebenso wäre es denkbar, eine Umfrage unter allen Mitarbeiterinnen und Mitarbeitern der Organisationseinheit, deren Arbeitsaufgaben mit Daten zu tun haben, zu erstellen. In dieser Umfrage sollten die Beteiligten nach dem Zustand vor und nach der Einführung des SDO Auskunft geben.

6.6 Vermeidung dunkler Daten durch optimierten Datenlebenszyklus

Letztendlich sollen alle vorgestellten Konzepte wegführen vom heutigen Datenlebenszyklus, dem dunkle Daten inhärent sind (wie in Kapitel 3.3 in Abbildung 3.5 dargestellt) hin zu einem optimalen Datenlebenszyklus. Die Optimierung wird dabei erreicht durch das Metadatenmodell *EngMeta*, die automatisierte Metadatenerfassung, die in den Arbeitsprozess integriert ist, die Rolle des *Scientific Data Officers* sowie der Entscheidungskriterien über Dauer der Speicherung von Daten. Diese Optimierungen sollen dunkle Daten verhindern, vermeiden oder zumindest eindämmen.

Das Metadatenmodell wurde in Kapitel 6.2 evaluiert und entspricht qualitativ den in (Bruce und Hillmann, 2004) und (NISO, 2007) aufgestellten Kriterien für gute Metadaten. Dies wird erreicht, indem *EngMeta* bekannte Metadatenstandards und Normdateien nutzt, durch XSD interoperabel gestaltet ist, fachspezifische, prozessspezifische Erhaltungsmetadaten und deskriptive Informationen enthält sowie dokumentiert und frei verfügbar ist. Ebenso ist das Metadatenmodell prinzipiell in allen Fachbereichen anwendbar, in denen mit Computermethoden Simulation durchgeführt wird, die auf kontrollierten und gemessenen Variablen sowie auf Parametern beruhen. Indem dieses Metadatenmodell auf Datensätze der Simulationswissenschaften angewendet wird, können dunkle Daten durch fehlende Metadatenannotation verhindert und so die Nachnutzbarkeit der Daten im Sinne der FAIR-Prinzipien ermöglicht werden.

Die Metadatenerfassung entlastet hierbei die Wissenschaftlerinnen und Wissenschaftler. Wie in Kapitel 1.3.2 ausgeführt ist der Mehraufwand eine der größten Hürden zur Metadatenannotation. Dieser Mehraufwand wird durch die automatisierte Metadatenerfassung wesentlich reduziert. Sie lässt sich allein durch das Schreiben von Konfigurationsdateien an spezifische Datenformate anpassen, die innerhalb gleichartiger Simulationscodes nur einmal erstellt werden müssen. Dabei lassen sich insbesondere technische, fachspezifische und prozessspezifische Metadaten automatisiert erfassen, wobei sich für die deskriptiven Metadaten Vorlagen schreiben lassen, die für viele Erfassungsvorgänge gelten können. Wie in 6.3 gezeigt wurde, ist die Erfassung auf den getesteten Systemen im Großrechnerumfeld (*Cray XC40*, *Cray URIKA*, *bwUniCluster*) lauffähig. Die Dauer der Erfassung liegt selbst für große Auswertungsdateien im Bereich von Sekunden, so dass der zeitliche Aufwand den Forschungsprozess selbst nicht verzögert.

Die Integration in den Arbeitsprozess wurde in Kapitel 6.4 evaluiert. Hier wurde festgestellt, dass sich die Metadatenerfassung einfach nach dem Simulationslauf als Epilog-Skript ausführen lässt. Darüber hinaus können mittels der DaRUS-App, daran anschließend, die Daten direkt automatisiert in ein Repositorium geladen werden. Die automatisierte Metadatenerfassung im Zusammenhang mit dem Einspielen der Daten in ein Repositorium leistet so einen wesentlichen Beitrag zur Reduzierung bzw. Verhinderung dunkler Daten. Dunkle Daten werden dadurch verhindert, dass

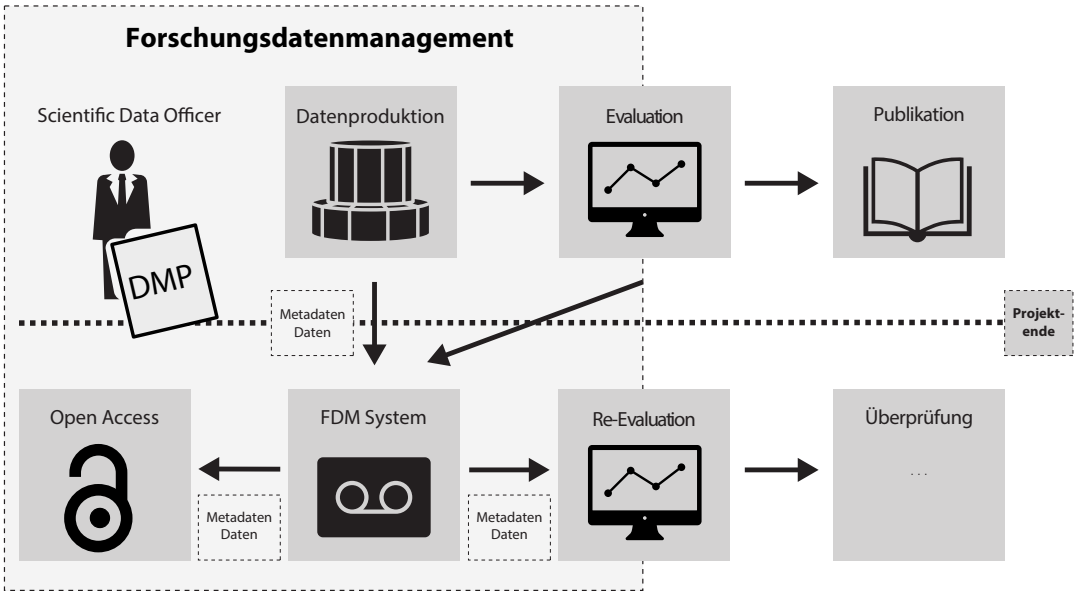

Abbildung 6.8: Optimierter Datenlebenszyklus. Vorab veröffentlicht in (Schembera und Bönisch, 2017).

die Daten durch die automatisierte Erfassung leicht mit Metadaten annotiert und in ein durchsuchbares Repositorium eingegliedert werden können. Dies verhindert sowohl dunkle Daten durch fehlende Metadatenannotation als auch dunkle Daten durch de-registrierte Nutzerinnen und Nutzer, da veröffentlichte Daten in einem Repositorium der Allgemeinheit zugänglich sind. Waren das Metadatenmodell sowie die -erfassung und Arbeitsprozessintegration vor allem Mittel zur Reduzierung dunkler Daten durch fehlende Metadatenannotation, verhindert der Scientific Data Officer insbesondere dunkle Daten durch de-registrierte oder inaktive Nutzerinnen und Nutzer. Durch seine Tätigkeit der Sichtbarmachung von Daten und das gezielte Einsetzen von Datenmanagementplänen verhindert dunkle Daten von vornherein. Dies wird erreicht durch Prozesse, wie sie in Kapitel 5.4 beschrieben werden, nämlich das periodische Checken des Dateninventars, die Unterstützung der Metadatenannotation sowie die Multiplikatorfunktion des SDO. Sollten dunkle Daten gefunden werden, richtet er sich nach den Prozessen, die in Form der Entscheidungskriterien in Kapitel 5.5 aufgestellt wurden, nimmt dunkle Daten unter seine Obhut oder löscht sie in Übereinstimmung mit evtl. bestehenden Verträgen oder rechtlichen Grundlagen.

In Abbildung 6.8 ist der optimierte Datenlebenszyklus schematisch nach der Implementierung der in der Dissertation vorgestellten Konzepte wiedergegeben. Dabei werden die Daten und Metadaten nach Generierung und Auswertung in ein Repositorium gespielt, sodass sie nicht dunkel werden können. Ein Scientific Data Officer überwacht hierbei die Prozesse und hat die Verantwortung. Ein Datenmanagementplan regelt das Datenhandling von vornherein.

Kapitel 7

Fazit

7.1 Zusammenfassung

Die grundlegende Motivation für geregeltes Forschungsdatenmanagement im Höchstleistungsrechnen bzw. in den computergestützten Ingenieurwissenschaften stellt das Vorkommen von dunklen Daten und deren Vermeidung dar. Dazu musste zu Beginn des Dissertationsvorhabens der Begriff der dunklen Daten für das Höchstleistungsrechnen definiert werden. In Kapitel 3 wurde gezeigt, dass auch in diesem Bereich – entgegen der eigentlichen Definition von Bryan Heidorn – dunkle Daten entstehen können. Die ursprüngliche Definition von Heidorn adressiert dunkle Daten im *long tail of science*, dem er schlechtes Datenmanagement unterstellt. In Kapitel 3.1 konnte allerdings gezeigt werden, dass die von Heidorn für den *long tail* ermittelten Charakteristika, nämlich dass die Daten und deren Management äußerst heterogen, ohne Standards, individuell verwaltet und selten nachgenutzt seien, auch für das Höchstleistungsrechnen als Vertreterin des *head* bzw. von Big Data gelten. Dunkle Daten zeigen sich als dem heutigen Datenlebenszyklus inhärent (siehe Kapitel 3.3 in Abbildung 3.5) und rühren im HPC insbesondere von fehlender Metadatenannotation als auch von Inaktivität der Nutzerinnen und Nutzern her. Schließlich wurden dunkle Daten im Bandspeichersystem des HLRS konkret nachgewiesen und in Kapitel 3.2 mit Statistiken in ihrer zeitlichen Entwicklung belegt. Da eine Metrik für dunkle Daten durch fehlende Metadaten noch aussteht, wurden letztlich nur die dunklen Daten durch Inaktivität festgestellt, deren Anteil sich über die letzten 5 Jahre um ca 3 % bewegt. Der Anteil der dunklen Daten durch fehlende Metadaten dürfte weitaus höher liegen – die vom DWD angegebenen 50 % (Heene u. a., 2016) dürften in den computergestützten Ingenieurwissenschaften bzw. im HPC im Allgemeinen als untere Grenze angesehen werden. Den Begriff der dunklen Daten auf das Höchstleistungsrechnen zu beziehen und nachzuweisen, dass dunkle Daten auch in persistenten Festspeichern entlang der kompletten Speicherhierarchie – und nicht wie in der originalen Definition nur durch portable Datenträger – entstehen, ist ein wesentlicher wissenschaftlich-analytischer Beitrag der Dissertation. Darüber hinaus wurde festgestellt, dass dunkle Daten durch fehlende Metadaten auch im HPC einen gewaltigen Anteil einnehmen, der jedoch nicht exakt quantifiziert, sondern nur abgeschätzt werden konnte.

Um das Aufkommen dunkler Daten zu verhindern, ist Metadatenannotation unabdingbar und stellt das erste Ziel von Forschungsdatenmanagement dar. Hierzu wurde das Metadatenmodell *EngMeta* entwickelt, welches über das Verständnis von Metadaten als „Daten über Daten“ weit hinausgeht. Metadaten sind laut (Greenberg, 2003) dagegen eine strukturierte Beschreibung der Merkmale eines Objekts, aber sie beinhalten auch die implizite Unterstützung des Arbeitsprozesses, beispielsweise durch Interpretationshilfen. Metadaten haben schließlich die Aufgabe, den Objektbezug herzustellen. Demnach darf ein Metadatenmodell zur Beschreibung von Forschungsdaten aus den Ingenieurwissenschaften nicht nur deskriptive Metadaten wie z.B. Autorschaft und Jahr der Erzeugung enthalten, sondern muss den komplexen Forschungsprozess als Ganzes beschreiben. Dies umfasst neben den technischen Metadaten, wie den Dateieigenschaften, insbesondere prozessspezifische und fachspezifische Metadaten. Die Prozessmetadaten erfüllen dabei im Wesentlichen die Aufgabe, nachvollziehbare Forschung zu garantieren, was durch die Aufnahme von Metadaten erreicht wird, welche die Modellimplementierung und ihre Ausführung auf Rechnersystemen betreffen. Die fachspezifischen Metadaten müssen alle Eigenschaften des Modellbildungsprozesses als solchem erfassen, also z.B. Eigenschaften des Zielsystems und seines Modells. Die Entwicklung von *EngMeta* wurde in Kapitel 5.2 nachgezeichnet. Hierzu wurde von einem Objektmodell ausgegangen, welches die zentralen Punkte des Simulations- und Modellbildungsprozesses in den computergestützten Ingenieurwissenschaften aufnimmt. Dann wurden die Objekte und ihre relevanten Eigenschaften expliziert. Andere Metadatenstandards wurden auf ihre Anwendbarkeit hin geprüft, allerdings stellte sich bei diesem Schritt heraus, dass keines der untersuchten Metadatenmodelle den Anforderungen der computergestützten Ingenieurwissenschaften genügt. Schließlich wurde *EngMeta* als Metadatenmodell unter Berücksichtigung der Standards *DataCite*, *CodeMeta*, *ExptML* sowie *PREMIS* erstellt. Da viele neue Metadatenentitäten eingeführt wurden, kann es nicht mehr nur als Anwendungsprofil verstanden werden. Es wurde als XML-Schema implementiert und steht zur Nutzung bereit.

In der Anforderungsanalyse zeigte sich, dass manuelle Metadatenannotation für die Wissenschaftlerinnen und Wissenschaftler eine große Hemmschwelle zur Nutzung von Metadatenprodukten darstellt. Darum wurde innerhalb des Dissertationsvorhabens das Konzept für eine automatisierte Metadatenerfassung vorgeschlagen und implementiert, welches in Kapitel 5.3 dargestellt ist. Diese Metadatenerfassung ist dabei insofern generisch, als dass sie über die Erstellung von Konfigurationsdateien auf neue Simulationscodes anpassbar ist. Dabei können jegliche textuell kodierten Informationen der Form *Schlüssel, Trenner, Wert* aus den Ausgabedateien ausgelesen und in *EngMeta* oder ein anderes Metadatenformat umgewandelt werden. Es wurde neben einer nativen Variante, die die Java Scanner API nutzt, auch eine parallele Variante mit dem Apache Spark Datenanalyse-Framework implementiert. Diese Variante soll insbesondere das Analysieren von sehr großen Datenmengen beschleunigen.

Im Verlauf des Forschungsvorhabens erwies sich, dass Metadaten ohne die entsprechenden unterstützenden Prozesse (Edwards u. a., 2011) und ohne „people power“ (Mattmann, 2013) wertlos sind, da sich einige Probleme nicht auf technischer Ebene lösen lassen. Diese Sichtweise wird von den erweiterten FAIR-Prinzipien (EU, 2018) bestärkt. Daher wurde in Kapitel 5.4 ein spezifischer, auf das Höchstleistungsrechnen zugeschnittener Datenkurator vorgeschlagen, nämlich der *Scientific Data Officer*, und auf die Metadatenprodukte bezogen. Dieser hat die Aufgabe, Metadatenannotation zu unterstützen, dunkle Daten aufzuspüren und die Verantwortung dafür zu übernehmen. Das Ziel seiner Tätigkeit ist die Sichtbarmachung von Daten durch Metadaten und die Übernahme und Entscheidung über die Speicherung von dunklen Daten. Hierfür wurden im Rahmen der Dissertation Entscheidungskriterien entwickelt, welche Daten aufbewahrt werden müssen und welche nicht. Diese Entscheidungskriterien wurden in Kapitel 5.5 für Daten als Grundlage von Publikationen, Daten ohne Publikation und dunkle Daten entwickelt. Die Rolle des SDO und die Entscheidungskriterien sind allerdings nicht auf den Bereich des Höchstleistungsrechnens beschränkt, sondern wurden als Anwendungsfall an diesem entwickelt.

Schließlich wurden die Konzepte in Kapitel 6 einer Evaluation unterzogen. Bzgl. *EngMeta* zeigte sich, dass das Metadatenmodell die qualitativen Kriterien von (Bruce und Hillmann, 2004) und (NISO, 2007) für ein gutes Metadatenmodell erfüllt. Ebenso entspricht es den von FAIR aufgestellten Anforderungen an Metadaten. Eine quantitative Evaluation konnte nicht durchgeführt werden, da die statistische Basis fehlt – *EngMeta* findet sich noch nicht im breiten Einsatz. Neben der Güte des Metadatenmodells wurde seine Adaptierbarkeit geprüft. *EngMeta* zeigt sich als für die auf computergestützter Simulation aufbauenden Wissenschaften als universell einsetzbar, da es Eigenschaften wie kontrollierte Variablen, Parameter, Software, Rechnerumgebung, die zeitlichen und räumlichen Eigenschaften einer Simulation usw. abbildet, die allen Fachbereichen gemein sind. Die automatisierte Metadatenerfassung wurde für die drei Datenformate GROMACS, EAS3 und NetCDF getestet. Die Erfassung von Metadaten aus diesen drei Simulationscodes ist möglich, jedoch lassen sich weitaus weniger Metadaten erfassen, als zunächst angenommen. Dies rührt daher, dass die Ausgabe der Simulationscodes weniger Informationen als erwartet mitliefert. Die Metadatenerfassung wurde auch einer Performanceanalyse unterzogen. Diese hat zum Ergebnis, dass für den typischen Anwendungsfall, bei dem Logdateien der Simulationscodes im Bereich von Megabytes liegen, die native Variante ausreichend ist und die Erfassung in wenigen Sekunden durchgeführt werden kann. Die parallele Variante mit Spark lohnt sich erst ab Datensätzen in der Größe von 23 MB. Die Metadatenerfassung wurde auch für die verschiedenen Rechnerumgebungen getestet und lässt sich durch die plattformunabhängige Implementierung mittels Java auf allen Systemen, auf denen Java 1.8 verfügbar ist, ausführen. Die Integration der Erfassung in die Großrechnerumgebung und das Zusammenspiel wurde evaluiert. Es konnte gezeigt werden, dass sich der Prozess *Ablauf der Simulation – Erfassung der Metadaten – Einspielen der Daten und Metadaten in ein Repositorium* vollständig automatisieren lässt.

7.2 Diskussion und Ausblick

Dass heutiges Forschungsdatenmanagement in den computergestützten Ingenieurwissenschaften an fehlender Metadatenannotation, fehlenden Anreizen und mangelnder Integration in den Prozess krankt, war die Ausgangslage der Dissertation. Sie definiert die in dieser Dissertation bearbeitete Problemstellung und ist gleichzeitig der Ansatzpunkt zur Optimierung von Forschungsdatenmanagement im untersuchten Fachbereich.

Die eingangs formulierte Hypothese, dass der *Zweck* des gelingenden Forschungsdatenmanagements sich durch die Wahl der entsprechenden technischen und vor allem auch organisatorischen *Mittel* realisiert, wurde innerhalb der Dissertation verifiziert. Dies geschah durch die Erstellung von *EngMeta* als Metadatenprodukt und der automatisierten Metadatenerfassung als unterstützendem Prozess, der die Hemmschwelle zur Metadatenannotation deutlich senken kann. Dabei wurden diese als Mittel erstellt, um dem Zweck der Optimierung des Arbeitsprozesses und damit FAIRem Forschungsdatenmanagement zu dienen. Diese Mittel sollen intrinsische Anreize zum Forschungsdatenmanagement verstärken bzw. eine Arbeitsprozessintegration erst ermöglichen. In der Evaluation zeigte sich, dass insbesondere die automatisierte Metadatenerfassung ein geeignetes Mittel zur Optimierung ist, da es den Aufwand der Metadatenannotation (nach erstmaliger Anpassung der Erfassung an den jeweiligen Fall) wesentlich vermindert und für viele Simulationscodes einsetzbar ist. Diese technischen Mittel mussten um nicht-technische ergänzt werden, da sich einerseits nicht alle Probleme rein technisch lösen lassen und dies andererseits durch Anforderungen an FAIRes Forschungsdatenmanagement vorgegeben ist. Die Notwendigkeit des *Scientific Data Officers* entspringt aus den Charakteristika der dunklen Daten selbst: Weil dunkle Daten unsichtbar und nicht mehr zugeordnet sind, sind Verantwortlichkeiten unklar – sie können nur über manuelle Intervention und verbindliche Entscheidungen geklärt werden, die außerhalb von der Bewertung durch Algorithmen liegen.

Entlang des Forschungsvorhabens wurden Fragen aufgeworfen, die – aus zeitlichen Gründen oder weil sie den Umfang der Arbeit sprengen würden – nicht innerhalb des Rahmens der Dissertation thematisiert oder besprochen werden konnten.

Da dunkle Daten hier das erste Mal für das Höchstleistungsrechnen definiert und diskutiert wurden, stehen für diesen Themenbereich noch einige Fragen offen. In der Dissertation konnten dunkle Daten durch Inaktivität im Bandspeichersystem des HLRS nachgewiesen werden. Von hohem Interesse wäre auch eine Statistik bzw. eine genaue Abschätzung dunkler Daten durch fehlende Metadatenannotation, wie sie für den DWD in (Heene u. a., 2016) durchgeführt wurde. Dies setzt selbstverständlich eine Metrik voraus, mit der bestimmt werden kann, *wann genau* Daten als annotiert gelten und wann nicht. Strikte Kriterien könnten beispielsweise verlangen, dass die Daten durch ein Metadatenschema wie *EngMeta* o.ä. beschrieben werden.

Weniger strikte Kriterien könnten eine beliebige Datendokumentation verlangen, die einer Konvention entspricht, beispielsweise der CF-Konvention von NetCDF. Sind diese Kriterien eindeutig, könnten automatisiert die Datenbestände durchsucht und festgestellt werden, welcher prozentuale Anteil der Daten als dunkel markiert würde. In diesem Zusammenhang ließe sich auch über *graue Daten* als Metapher für teil-annotierte Daten denken. Daran anschließend wäre eine Bestandserfassung zu dunklen Daten wünschenswert, die sich über mehrere Hoch- und Höchstleistungsrechenzentren erstreckt, da nur so der komplette quantitative Umfang des Problems erfasst werden kann.

Neben diesen statistischen Folgearbeiten stellt sich auch die Frage nach der Rolle von Dateisystemen im Bezug auf dunkle Daten. Obwohl parallele Dateisysteme die heißen Daten enthalten und am HLRS beispielsweise durch den Workspace-Mechanismus vor dunklen Daten durch Inaktivität geschützt scheinen, können diese hier auftreten. Hier wäre zu klären, was Dateisysteme dazu beitragen können, dunkle Daten entweder zu verhindern oder zu erkennen.

Vom wissenschaftstheoretischen und technikphilosophischen Standpunkt aus stellt sich die Frage, wie das Verhältnis von dunklen Daten zur Reproduzierbarkeit von Forschungsdaten genau bestimmt wird. Wie genau dunkle Daten zur Reproduzierbarkeitskrise beitragen, bleibt noch zu klären.

Anläßlich der Annahme, dass die maximale technische Integrationsdichte von elektrotechnischen Komponenten in den nächsten Jahren erreicht werden wird und die Moorsche Gesetzmäßigkeit vor ihrem Ende steht (Courtland, 2016), gibt es viele kritische Implikationen bezogen auf die Datenproduktion und das Forschungsdatenmanagement. In diesem Rahmen wäre beispielsweise zu prüfen, wie sich das Datenwachstum im HPC nach 2020 verhalten wird, also eine Aktualisierung der Studie, wie sie in (Hick, 2010) für die gegenwärtige Dekade durchgeführt wurde. Die weitaus spannendere Frage ist allerdings, ob Forschungsdatenmanagement im Zuge dieser Erkenntnis grundsätzlich an Wichtigkeit gewinnen wird. In Zukunft könnte es vermehrt um die Nachnutzung von bereits berechneten Daten gehen, wofür Metadatendokumentation und Unterstützungsprozesse, wie sie in der Dissertation vorgeschlagen wurden, unabdingbar sind.

Mit *EngMeta* wurde im Rahmen der Dissertation ein Metadatenmodell für die computergestützten Ingenieurwissenschaften vorgeschlagen. Dieser Ansatz konnte im Promotionsvorhaben nur qualitativ evaluiert werden, da das Metadatenmodell noch nicht breit genutzt wird. Eine quantitative Analyse ist erst möglich, wenn ausreichend viele Datensätze mit *EngMeta* beschrieben sind. Eine quantitative Evaluation kann z.B. mit dem *Metadata Quality Evaluation Model* (MQEM) durchgeführt werden (Gavrilis u. a., 2015).

Der Ansatz zur automatisierten Metadatenerfassung wurde für die Dissertation zunächst prototypisch implementiert. Hier gäbe es viele Erweiterungsmöglichkeiten bzgl. der Implementierung, beispielsweise bessere Trenner-Funktionen. Ebenso könnte überlegt werden, ob und inwiefern der Ansatz von Machine Learning profitieren kann.

Es zeigte sich bei der Analyse, dass sich weitaus weniger Metadaten als erhofft erfassen lassen, weil diese schlichtweg nicht verfügbar sind, sofern keine Kontrolle über die Ausgaben der Simulationscodes selbst möglich ist. Die Frage wäre zu klären, wie sich die Situation verbessern ließe und wo hier die Ansätze zu suchen sind.

Abschließend lässt sich feststellen, dass das Forschungsdatenmanagement in den computergestützten Ingenieurwissenschaften und im Höchstleistungsrechnen noch einen weiten Weg vor sich hat, um als etabliert gelten zu können, obwohl die technologischen Mittel bereit stehen. Grundsätzlich müssen die Vorteile, die mit Metadatenannotation einhergehen, den Zusatzaufwand rechtfertigen und eine konkrete Aufwandsersparnis oder eine erhöhte Reputation für die einzelnen Wissenschaftlerinnen und Wissenschaftler mit sich bringen. Hierzu ist aber auch die Anerkennung von Forschungsdatenmanagement als der wissenschaftlichen Praxis inhärente Tätigkeit fundamental wichtig. Wird Forschungsdatenmanagement nur als notwendiges Übel gesehen, verschlimmert sich die Lage bzgl. dunkler Daten lediglich. Es müssen neben den intrinsischen Anreizen, die Thema dieser Dissertation waren, auch extrinsische geschaffen werden, die das Forschungsdatenmanagement und die Metadatenannotation als obligatorisch setzen. Dies kann nur mittels der Anerkennung von Forschungsdatenmanagement als wesentlichem Teil der wissenschaftlichen Praxis von Verlagen, Institutionen, Geldgebern und schlussendlich von Wissenschaftlerinnen und Wissenschaftlern selbst erreicht werden.

Anhang A

Appendix: Konfigurationsdateien Metadatenerfassung

A.1 Konfigurationsdatei für GROMACS-Daten

```
1  # Configuration file for Harvester
2
3  # The "code" key defines the general simulation package used
4  # This information can later be used by the harvester to take
       specific action
5  code=gromacs
6  data=trr
7
8  #This parameter determines if the data file should be checksummed.
       Possible values are: yes, no
9  checksum=no
10
11 #This parameter defines which result should be taken if multiple
       results are found. Possible values: first,last
12 matching=all
13
14
15
16 # MDKey,File,SearchKey,Delimiter,Semantics (if applicable)
17 ##
18 contact.name,usermd,lastname,=,
19 contact.givenName,usermd,firstname,=,
20 contact.familyName,usermd,lastname,=,
21 contact.address,usermd,address,=,
22 contact.affiliation.name,usermd,affiliation.name,=,
23 contact.affiliation.address,usermd,address,=,
24 contact.affiliation.email,usermd,email,=,
25 contact.email,usermd,email,=,
26 contact.id,usermd,contact.id,=,
27 contact.role,usermd,contact.role,=,
28
29 contact.name,usermd,lastname2,=,2
```

```
30 contact.givenName,usermd,firstname2,=,2
31 contact.familyName,usermd,lastname2,=,2
32 contact.address,usermd,address2,=,2
33 contact.affiliation.name,usermd,affiliation.name2,=,2
34 contact.affiliation.address,usermd,address2,=,2
35 contact.affiliation.email,usermd,email2,=,2
36 contact.email,usermd,email2,=,2
37 contact.id,usermd,contact.id2,=,2
38 contact.role,usermd,contact.role2,=,2
39
40
41 # Creator
42
43 creator.name,usermd,creator.name,=,
44 creator.givenName,usermd,creator.firstname,=,
45 creator.familyName,usermd,creator.lastname,=,
46 creator.address,usermd,creator.address,=,
47 creator.affiliation.name,usermd,creator.affiliation.name,=,
48 creator.affiliation.address,usermd,creator.affiliation.address,=,
49 creator.email,usermd,creator.email,=,
50 creator.role,usermd,creator.role,=,
51
52 # Contributor
53
54 contributor.name,usermd,contributor.name,=,
55 contributor.givenName,usermd,contributor.firstname,=,
56 contributor.familyName,usermd,contributor.lastname,=,
57 contributor.address,usermd,contrinutor.address,=,
58 contributor.affiliation.name,usermd,contributor.affiliation.name,=,
59 contributor.affiliation.address,usermd,contributor.affiliation.
       address,=,
60 contributor.affiliation.email,usermd,contributor.affiliation.email
       ,=,
61 contributor.affiliation.id,usermd,contributor.affiliation.id,=,
62 contributor.email,usermd,contributor.email,=,
63 contributor.id,usermd,contributor.id,=,
64 contributor.role,usermd,contributor.role,=,
65
66 # Project
67
68 project.value,usermd,project.value,=
69 project.level,usermd,project.level,=
70
71 # Funding Information
72
73 fundingReference.funderName,usermd,fundingReference.funderName,=
74 fundingReference.funderIdentifier,usermd,fundingReference.
       funderIdentifier,=
```

```
75 fundingReference.awardNumber,usermd,fundingReference.awardNumber,=,
76 fundingReference.awardNumber.URI
77 fundingReference.awardTitle
78
79 # Worked
80
81 worked,usermd,worked,=
82
83 # Worked Note
84
85
86 workedNote,usermd,workedNote,=
87
88 # Title
89
90 title,usermd,title,=
91 title.titleType
92 title.lang
93
94
95 # Description
96
97 description,usermd,description,=
98 description.descriptionType
99 description.lang
100
101 # Resource Type
102
103 resourceType,usermd,resourceType,=,
104 resourceType.resourceTypeGeneral
105
106 # Keywords
107
108 keywords.keyword
109 keywords.keyword.vocabulary
110 keywords.keyword.vocabularyURL
111
112 # Subject
113
114 subjects.subject,usermd,subject,=,
115 subjects.subject.subjectScheme
116 subjects.subject.schemeURI
117 subjects.subject.valueURI
118 subjects.subject.lang
119
120 # Date
121
122 dates.date,usermd,dates.date,=
```

```
123 dates.date.dateType
124
125 # Version
126
127 version,usermd,version,=
128
129 # Creation Mode
130 mode,usermd,mode,=
131
132 # Measured Variables
133 #
134 # Measured or simulated variables.
135 measuredVariable.name
136 measuredVariable.symbol
137 measuredVariable.value
138 measuredVariable.encoding
139 measuredVariable.error
140
141 # Controlled Variables
142 controlledVariable.name
143 controlledVariable.symbol
144 controlledVariable.value
145 controlledVariable.encoding
146 controlledVariable.error
147
148 controlledVariable.name,usermd,var1.name,=,1
149 controlledVariable.value,mdp,ref_t,=,1
150 controlledVariable.name,usermd,var2.name,=,2
151 controlledVariable.value,mdp,tcoupl,=,2
152 controlledVariable.name,usermd,var3.name,=,3
153 controlledVariable.value,mdp,ref_p,=,3
154 controlledVariable.name,usermd,var4.name,=,4
155 controlledVariable.value,mdp,pcoupl,=,4
156 provenance.processingStep.executionCommand,mdp,gmx_mpi grompp,na,1
157 provenance.processingStep.executionCommand,log,gmx_mpi mdrun,na,2
158 provenance.processingStep.environment.compiler.name,log,g++ ,na,1
159 provenance.processingStep.environment.compiler.flags,log,C++
    compiler flags,:,1
160 provenance.processingStep.environment.nodes,job,nodes,=,
161 provenance.processingStep.environment.ppn,job,ppn,=,
162 provenance.processingStep.environment.cpu,log,Build CPU brand,:
163 system.temporalResolution.numberOfTimesteps,mdp,nsteps,=,
164 system.temporalResolution.interval,mdp,dt,=,
165
166 #execution command that was called in this processingStep
167 provenance.processingStep.executionCommand,mdp,gmx_mpi grompp,na,1
168 provenance.processingStep.executionCommand,log,gmx_mpi mdrun,na,2
169
```

```
170
171 # Instrument used in this processingStep
172 provenance.processingStep.instrument.name
173 provenance.processingStep.instrument.description
174 provenance.processingStep.instrument.type
175 provenance.processingStep.instrument.partnum.value
176 provenance.processingStep.instrument.partnum.vendor
177 provenance.processingStep.instrument.serialnum.value
178 provenance.processingStep.instrument.serialnum.vendor
179 provenance.processingStep.instrument.software
180 provenance.processingStep.instrument.software.os
181 provenance.processingStep.instrument.date
182
183 # Output file or stream, specified by (P)ID or link and checksum (
    optional)
184 provenance.processingStep.output.id
185 provenance.processingStep.output.id.type
186 provenance.processingStep.output.id.scheme
187 provenance.processingStep.output.link
188 provenance.processingStep.output.checksum
189 provenance.processingStep.output.checksum.type
190
191
192
193
194 # Computing environment of this processingStep
195 provenance.processingStep.environment.name
196 provenance.processingStep.environment.compiler.name,log,g++ ,na,1
197 provenance.processingStep.environment.compiler.flags,log,C++
    compiler flags,:,1
198 provenance.processingStep.environment.nodes,job,nodes,=,
199 provenance.processingStep.environment.ppn,job,ppn,=,
200 provenance.processingStep.environment.cpu,log,Build CPU brand,:
201
202 provenance.processingStep.environment.compiler.name,log,gcc ,na,2
203 provenance.processingStep.environment.compiler.flags,log,C compiler
     flags,:,2
204
205
206 provenance.processingStep.order
207
208 # Size
209
210 size
211
212 # System
213
214 system.phase
```

```
215 system.phase.component.name
216 system.phase.component.smilesCode
217 system.phase.component.IUPAC
218 system.phase.component.quantity
219 system.phase.component.unit
220 system.phase.component.forcefield.name
221 system.phase.component.forcefield.parameter.name
222 system.phase.component.forcefield.parameter.symbol
223 system.phase.component.forcefield.parameter.value
224 system.phase.component.forcefield.parameter.encoding
225 system.phase.component.forcefield.parameter.error
226 system.component.name
227 system.component.smilesCode
228 system.component.IUPAC
229 system.component.quantity
230 system.component.unit
231 system.component.forcefield.name
232 system.component.forcefield.parameter.name
233 system.component.forcefield.parameter.symbol
234 system.component.forcefield.parameter.value
235 system.component.forcefield.parameter.encoding
236 system.component.forcefield.parameter.error
237 system.grid.file.id
238 system.grid.file.id.type
239 system.grid.file.id.scheme
240 system.grid.file.link
241 system.grid.file.checksum
242 system.grid.file.checksum.type
243 system.grid.countCells
244 system.grid.countBlocks
245 system.grid.targetCores
246 system.grid.countX
247 system.grid.countY
248 system.grid.countZ
249 system.grid.unit
250 system.grid.distance
251 system.grid.scalingFormula
252 system.grid.point.positionX
253 system.grid.point.positionY
254 system.grid.point.positionZ
255 system.temporalResolution.numberOfTimesteps,mdp,nsteps,=,
256 system.temporalResolution.interval,mdp,dt,=,
257 system.temporalResolution.unit
258 system.temporalResolution.timestep
259
260 # Storage information
261 storage.contentLocation.value
262 storage.contentLocation.link
```

```
storage.storageMedium
storage.storageMedium.URI

# Format information
format.formatDesignation.formatName
format.formatDesignation.formatVersion
format.formatRegistry.name
format.formatRegistry.key
format.formatRegistry.role

# The PID of the dataset
pid
pid.type
pid.scheme

# Checksum of the dataset
checksum
checksum.algorithm

# Rights of the dataset
rightsStatement.copyrightInformation.note
rightsStatement.copyrightInformation.status
rightsStatement.copyrightInformation.countryCode

#Context objects
context.referencePublication.id
context.referencePublication.id.type
context.referencePublication.id.scheme
context.referencePublication.citation
context.referencePublication.url
context.relatedResource.id
context.relatedResource.id.type
context.relatedResource.id.scheme
context.relatedResource.link
context.relatedResource.checksum
context.relatedResource.checksum.scheme
context.relatedIdentifier
context.relatedIdentifier.type
context.relatedIdentifier.scheme
```

Listing A.1: Für die Evaluation genutzt Konfigurationsdatei zur Metadatenerfassung von GROMACS-Simulationsdaten.

A.2 Konfigurationsdatei für EAS3-Daten

```
# Configuration file for Harvester

```

```
3  # The "code" key defines the general simulation package used
4  # This information can later be used by the harvester to take
       specific action
5  code=ns3d
6  data=eas
7
8  #This parameter determines if the data file should be checksummed.
       Possible values are: yes, no
9  checksum=no
10
11 #This parameter defines which result should be taken if multiple
       results are found. Possible values: first,last
12 matching=first
13
14 # MDKey,File,SearchKey,Delimiter,Semantics (if applicable)
15 ##
16 contact.name,usermd,contact1.lastname,=,1
17 contact.givenName,usermd,contact1.firstname,=,1
18 contact.familyName,usermd,contact1.lastname,=,1
19 contact.address,usermd,contact1.address,=,1
20 contact.affiliation.name,usermd,contact1.affiliation.name,=,1
21 contact.affiliation.address,usermd,contact1.address,=,1
22 contact.affiliation.email,usermd,contact1.email,=,1
23 contact.email,usermd,contact1.email,=,1
24 contact.id,usermd,contact1.id,=,1
25 contact.role,usermd,contact1.role,=,1
26
27
28
29 contact.name,usermd,contact2.name,=,3
30 contact.givenName,usermd,contact2.firstname,=,3
31 contact.familyName,usermd,contact2.lastname,=,3
32 contact.address,usermd,contact2.address,=,3
33 contact.affiliation.name,usermd,contact2.affiliation.name,=,3
34 contact.email,usermd,contact2.email,=,3
35 contact.role,usermd,contact2.role,=,3
36
37
38
39 # Creator
40
41 creator.name,usermd,contact1.name,=,1
42 creator.givenName,usermd,contact1.firstname,=,1
43 creator.familyName,usermd,contact1.lastname,=,1
44 creator.address,usermd,contact1.address,=,1
45 creator.affiliation.name,usermd,contact1.affiliation.name,=,1
46 creator.affiliation.address,usermd,contact1.affiliation.address,=,1
47 creator.email,usermd,contact1.email,=,1
```

```
48 creator.role,usermd,contact1.role,=,1
49
50 # Contributor
51
52 contributor.name,usermd,contact2.name,=,1
53 contributor.givenName,usermd,contact2.firstname,=,1
54 contributor.familyName,usermd,contact2.lastname,=,1
55 contributor.address,usermd,contact2.address,=,1
56 contributor.affiliation.name,usermd,contact2.affiliation.name,=,1
57 contributor.affiliation.address,usermd,contact2.affiliation.address
    ,=,1
58 contributor.affiliation.email,usermd,contact2.affiliation.email,=,1
59 contributor.affiliation.id,usermd,contact2.affiliation.id,=,1
60 contributor.email,usermd,contact2.email,=,1
61 contributor.id,usermd,contact2.id,=,1
62 contributor.role,usermd,contact2.role,=,1
63
64 # Project
65
66 project.value,usermd,project.value,=,1
67 project.level,usermd,project.level,=,1
68
69 # Funding Information
70
71 fundingReference.funderName,usermd,fundingReference.funderName,=,1
72 fundingReference.funderIdentifier,usermd,fundingReference.
    funderIdentifier,=,1
73 fundingReference.awardNumber,usermd,fundingReference.awardNumber
    ,=,1
74 fundingReference.awardNumber.URI
75 fundingReference.awardTitle
76
77 # Worked
78
79 worked,usermd,worked,=,1
80
81 # Worked Note
82
83 workedNote,usermd,workedNote,=,1
84
85 # Title
86
87 title,usermd,title,=,
88 title.titleType
89 title.lang
90
91
92 # Description
```

```
93
94  description ,. i ,case_comment,=
95  description . descriptionType ,usermd , description . descriptionType ,=,
96  description . lang ,usermd , description . lang ,=,
97
98  # Resource Type
99
100 resourceType ,usermd , resourceType ,=,1
101 resourceType . resourceTypeGeneral
102
103 # Keywords
104
105 keywords . keyword ,usermd , keyword1 ,=,1
106 keywords . keyword ,usermd , keyword2 ,=,2
107 keywords . keyword ,usermd , keyword3 ,=,3
108 keywords . keyword ,usermd , keyword4 ,=,4
109
110
111 # Subject
112
113 subjects . subject ,usermd , subject ,=,1
114 subjects . subject . subjectScheme
115 subjects . subject . schemeURI
116 subjects . subject . valueURI
117 subjects . subject . lang ,usermd , subject . lang ,=,1
118
119 subjects . subject ,usermd , subject2 ,=,2
120 subjects . subject . lang ,usermd , subject2 . lang ,=,2
121
122 # Date
123
124 dates . date ,usermd , dates . date ,=,1
125 dates . date . dateType
126
127 # Version
128
129 version ,usermd , version ,=,
130
131 # Creation Mode
132
133 mode, usermd , mode,=,1
134
135 boundaryCondition , ns3d . i , bc_marker_1 ,=,1
136 boundaryCondition , ns3d . i , bc_marker_2 ,=,2
137 boundaryCondition , ns3d . i , bc_marker_3 ,=,3
138 boundaryCondition , ns3d . i , bc_marker_4 ,=,4
139 boundaryCondition , ns3d . i , bc_marker_5 ,=,5
140 boundaryCondition , ns3d . i , bc_marker_6 ,=,6
```

```
141
142 grid_x_scheme,ns3d.i,spatial_scheme_x,=,1
143 grid_y_scheme,ns3d.i,spatial_scheme_y,=,1
144 grid_z_scheme,ns3d.i,spatial_scheme_z,=,1
145
146 grid_nx,log,Grid points nx \(Rank 0\),space,1
147 grid_ny,log,Grid points ny \(Rank 0\),space,1
148 grid_nz,log,Grid points nz \(Rank 0\),space,1
149 grid_trafo,log,Grid transformation,space,1
150
151
152 # Controlled Variables
153
154 controlledVariable.name,usermd,Mach,=,1
155 controlledVariable.name,usermd,Reynolds,=,7
156 controlledVariable.name,usermd,Prandtl,=,3
157 controlledVariable.name,usermd,time,=,4
158
159 controlledVariable.value,log,Mach,space,1
160 controlledVariable.value,log,Reynolds,space,7
161 controlledVariable.value,log,Prandtl,space,3
162 controlledVariable.value,log,time,space,4
163
164 # Provenance
165
166 provenance.processingStep.type,usermd,provenance.processingStep.
    type,=,
167
168 # The actor that is responsible for the processing step
169 provenance.processingStep.actor.name,usermd,creator.name,=
170 provenance.processingStep.actor.givenName,usermd,creator.firstname
    ,=
171 provenance.processingStep.actor.familyName,usermd,creator.lastname
    ,=
172 provenance.processingStep.actor.address,usermd,creator.address,=
173 provenance.processingStep.actor.affiliation.name,usermd,creator.
    affiliation.name,=
174 provenance.processingStep.actor.affiliation.address,usermd,creator.
    affiliation.address,=
175 provenance.processingStep.actor.email,usermd,creator.email,=
176 provenance.processingStep.actor.id,usermd,creator.id,=,
177 provenance.processingStep.actor.role,usermd,creator.role,=
178
179 # A date associated with the processingStep
180 provenance.processingStep.date,usermd,dates.date,=
181
182 # Definition of used method
```

```
183 provenance.processingStep.method.name.value,usermd,provenance.
    processingStep.method.name.value,=
184 provenance.processingStep.method.name.methodScheme
185 provenance.processingStep.method.name.schemeURI
186 provenance.processingStep.method.name.valueURI
187 provenance.processingStep.method.description
188 provenance.processingStep.method.parameter.name
189 provenance.processingStep.method.parameter.symbol
190 provenance.processingStep.method.parameter.value
191 provenance.processingStep.method.parameter.encoding
192 provenance.processingStep.method.parameter.error
193
194 # Error method. Same definition as for the general methods
195 provenance.processingStep.errormethod.name.value
196 provenance.processingStep.errormethod.name.methodScheme
197 provenance.processingStep.errormethod.name.schemeURI
198 provenance.processingStep.errormethod.name.valueURI
199 provenance.processingStep.errormethod.description
200 provenance.processingStep.errormethod.parameter.name
201 provenance.processingStep.errormethod.parameter.symbol
202 provenance.processingStep.errormethod.parameter.value
203 provenance.processingStep.errormethod.parameter.encoding
204 provenance.processingStep.errormethod.parameter.error
205
206
207 # Input file or stream, specified by (P)ID or link and checksum (
    optional)
208 provenance.processingStep.input.id
209 provenance.processingStep.input.id.type
210 provenance.processingStep.input.id.scheme
211 provenance.processingStep.input.link
212 provenance.processingStep.input.checksum
213 provenance.processingStep.input.checksum.type
214
215 # Software or code, that is used for the simulation.
216 provenance.processingStep.tool.name,ns3d.i,Typ,=,
217 provenance.processingStep.tool.contributor.name
218 provenance.processingStep.tool.contributor.givenName
219 provenance.processingStep.tool.contributor.familyName
220 provenance.processingStep.tool.contributor.address
221 provenance.processingStep.tool.contributor.affiliation
222 provenance.processingStep.tool.contributor.email
223 provenance.processingStep.tool.contributor.id
224 provenance.processingStep.tool.contributor.role
225 provenance.processingStep.tool.softwareVersion,log,\| Version,space
    ,
226 provenance.processingStep.tool.operatingSystem,log,Build OS/arch,:
227 provenance.processingStep.tool.programmingLanguage
```

```
228 provenance.processingStep.tool.SoftwareSourceCode.id
229 provenance.processingStep.tool.SoftwareSourceCode.id.type
230 provenance.processingStep.tool.SoftwareSourceCode.id.scheme
231 provenance.processingStep.tool.SoftwareSourceCode.link
232 provenance.processingStep.tool.SoftwareSourceCode.checksum
233 provenance.processingStep.tool.SoftwareSourceCode.checksum.type
234 provenance.processingStep.tool.SoftwareApplication.id
235 provenance.processingStep.tool.SoftwareApplication.id.type
236 provenance.processingStep.tool.SoftwareApplication.id.scheme
237 provenance.processingStep.tool.SoftwareApplication.link
238 provenance.processingStep.tool.SoftwareApplication.checksum
239 provenance.processingStep.tool.SoftwareApplication.checksum.type
240 provenance.processingStep.tool.codeRepository
241 provenance.processingStep.tool.license.note
242 provenance.processingStep.tool.license.terms
243 provenance.processingStep.tool.citation
244 provenance.processingStep.tool.referencedPublication.id
245 provenance.processingStep.tool.referencedPublication.id.scheme
246 provenance.processingStep.tool.referencedPublication.id.type
247 provenance.processingStep.tool.referencedPublication.citation
248 provenance.processingStep.tool.referencedPublication.url
249
250 # Exact execution command that was called in this processingStep
251
252 provenance.processingStep.executionCommand,log,gmx_mpi mdrun,line
253
254
255 # Instrument used in this processingStep
256 provenance.processingStep.instrument.name
257 provenance.processingStep.instrument.description
258 provenance.processingStep.instrument.type
259 provenance.processingStep.instrument.partnum.value
260 provenance.processingStep.instrument.partnum.vendor
261 provenance.processingStep.instrument.serialnum.value
262 provenance.processingStep.instrument.serialnum.vendor
263 provenance.processingStep.instrument.software
264 provenance.processingStep.instrument.software.os
265 provenance.processingStep.instrument.date
266
267 # Output file or stream, specified by (P)ID or link and checksum (
      optional)
268 provenance.processingStep.output.id
269 provenance.processingStep.output.id.type
270 provenance.processingStep.output.id.scheme
271 provenance.processingStep.output.link
272 provenance.processingStep.output.checksum
273 provenance.processingStep.output.checksum.type
274
```

```
275 # Computing environment of this processingStep
276 provenance.processingStep.environment.name
277 provenance.processingStep.environment.compiler.name,log,compiled
    with,na
278 provenance.processingStep.environment.compiler.flags,log,line,18
279 provenance.processingStep.environment.nodes,job,nodes,=,
280 provenance.processingStep.environment.ppn,job,ppn,=,
281 provenance.processingStep.environment.cpu,log,Build CPU brand,:
282
283 provenance.processingStep.order
284
285 # Size
286 #
287 # File size of the data.
288 size
289
290 # System
291
292 system.phase
293 system.phase.component.name
294 system.phase.component.smilesCode
295 system.phase.component.IUPAC
296 system.phase.component.quantity
297 system.phase.component.unit
298 system.phase.component.forcefield.name
299 system.phase.component.forcefield.parameter.name
300 system.phase.component.forcefield.parameter.symbol
301 system.phase.component.forcefield.parameter.value
302 system.phase.component.forcefield.parameter.encoding
303 system.phase.component.forcefield.parameter.error
304 system.component.name
305 system.component.smilesCode
306 system.component.IUPAC
307 system.component.quantity
308 system.component.unit
309 system.component.forcefield.name
310 system.component.forcefield.parameter.name
311 system.component.forcefield.parameter.symbol
312 system.component.forcefield.parameter.value
313 system.component.forcefield.parameter.encoding
314 system.component.forcefield.parameter.error
315 system.grid.file.id
316 system.grid.file.id.type
317 system.grid.file.id.scheme
318 system.grid.file.link
319 system.grid.file.checksum
320 system.grid.file.checksum.type
321 system.grid.countCells
```

```
322 system.grid.countBlocks,log,
323 system.grid.targetCores
324 system.grid.countX
325 system.grid.countY
326 system.grid.countZ
327 system.grid.unit
328 system.grid.distance
329 system.grid.scalingFormula
330 system.grid.point.positionX
331 system.grid.point.positionY
332 system.grid.point.positionZ
333 system.temporalResolution.numberOfTimesteps,ns3d.i,niter,=,
334 system.temporalResolution.interval,ns3d.i,dt,=,
335 system.temporalResolution.unit
336 system.temporalResolution.timestep
337
338 system.boundaryCondition.parameter.name,ns3d.i,bc_marker_1,=,1
339 system.boundaryCondition.parameter.name,ns3d.i,bc_marker_2,=,2
340 system.boundaryCondition.parameter.name,ns3d.i,bc_marker_3,=,3
341 system.boundaryCondition.parameter.name,ns3d.i,bc_marker_4,=,4
342 system.boundaryCondition.parameter.name,ns3d.i,bc_marker_5,=,5
343 system.boundaryCondition.parameter.name,ns3d.i,bc_marker_6,=,6
344
345 system.boundaryCondition.parameter.error,ns3d.i,wall_epsilon,=,1
346
347 system.grid.countCells,log,Total number of grid points,space,1
348 system.grid.countX,log,Grid points nx \(Rank 0\),space,1
349 system.grid.countY,log,Grid points ny \(Rank 0\),space,1
350 system.grid.countZ,log,Grid points nz \(Rank 0\),space,1
351
352 system.boundaryCondition.size
353 system.boundaryCondition.position
354 system.boundaryCondition.component.name
355 system.boundaryCondition.component.smilesCode
356 system.boundaryCondition.component.IUPAC
357 system.boundaryCondition.component.quantity
358 system.boundaryCondition.component.unit
359 system.boundaryCondition.component.forcefield.name
360 system.boundaryCondition.component.forcefield.parameter.name
361 system.boundaryCondition.component.forcefield.parameter.symbol
362 system.boundaryCondition.component.forcefield.parameter.value
363 system.boundaryCondition.component.forcefield.parameter.encoding
364 system.boundaryCondition.component.forcefield.parameter.error
365 system.boundaryCondition.parameter.name
366 system.boundaryCondition.parameter.symbol
367 system.boundaryCondition.parameter.value
368 system.boundaryCondition.parameter.encoding
369 system.boundaryCondition.parameter.error
```

```
370
371
372 # Storage information
373 storage.contentLocation.value
374 storage.contentLocation.link
375 storage.storageMedium
376 storage.storageMedium.URI
377
378 # Format information
379 format.formatDesignation.formatName
380 format.formatDesignation.formatVersion
381 format.formatRegistry.name
382 format.formatRegistry.key
383 format.formatRegistry.role
384
385
386 # The PID of the dataset
387 pid
388 pid.type
389 pid.scheme
390
391 # Checksum of the dataset
392 checksum
393 checksum.algorithm
394
395 # Rights of the dataset
396 rightsStatement.copyrightInformation.note
397 rightsStatement.copyrightInformation.status
398 rightsStatement.copyrightInformation.countryCode
399
400
401 #Context objects
402 context.referencePublication.id
403 context.referencePublication.id.type
404 context.referencePublication.id.scheme
405 context.referencePublication.citation
406 context.referencePublication.url
407 context.relatedResource.id
408 context.relatedResource.id.type
409 context.relatedResource.id.scheme
410 context.relatedResource.link
411 context.relatedResource.checksum
412 context.relatedResource.checksum.scheme
413 context.relatedIdentifier
414 context.relatedIdentifier.type
415 context.relatedIdentifier.scheme
```

Listing A.2: Für die Evaluation genutzt Konfigurationsdatei zur Metadatenerfassung von EAS3-Simulationsdaten.

A.3 Konfigurationsdatei für NetCDF-Daten

```
# Configuration file for Harvester

# The "code" key defines the general simulation package used
# This information can later be used by the harvester to take specific action
code=ccsm
data=nc

#This parameter determines if the data file should be checksummed. Possible values are: yes, no
checksum=no

#This parameter defines which result should be taken if multiple results are found. Possible values: first,last
matching=all

# MDKey,File,SearchKey,Delimiter,Semantics (if applicable)
##
contact.name,usermd,lastname,=,
contact.givenName,usermd,firstname,=,
contact.familyName,usermd,lastname,=,
contact.address,usermd,address,=,
contact.affiliation.name,cdl,institution,=,
contact.affiliation.address,usermd,address,=,
contact.affiliation.email,usermd,email,=,
contact.email,cdl,contact,=,
contact.id,usermd,contact.id,=,
contact.role,usermd,contact.role,=,


# Creator

creator.name,usermd,creator.name,=,
creator.givenName,usermd,creator.firstname,=,
creator.familyName,usermd,creator.lastname,=,
creator.address,usermd,creator.address,=,
creator.affiliation.name,usermd,creator.affiliation.name,=,
creator.affiliation.address,usermd,creator.affiliation.address,=,
creator.email,usermd,creator.email,=,
creator.role,usermd,creator.role,=,
```

```
38
39 # Contributor
40
41 contributor.name,usermd,contributor.name,=,
42 contributor.givenName,usermd,contributor.firstname,=,
43 contributor.familyName,usermd,contributor.lastname,=,
44 contributor.address,usermd,contrinutor.address,=,
45 contributor.affiliation.name,usermd,contributor.affiliation.name,=,
46 contributor.affiliation.address,usermd,contributor.affiliation.
       address,=,
47 contributor.affiliation.email,usermd,contributor.affiliation.email
       ,=,
48 contributor.affiliation.id,usermd,contributor.affiliation.id,=,
49 contributor.email,usermd,contributor.email,=,
50 contributor.id,usermd,contributor.id,=,
51 contributor.role,usermd,contributor.role,=,
52
53 # Project
54
55 project.value,cdl,project_id,=
56 project.level,usermd,project.level,=
57
58 # Funding Information
59
60 fundingReference.funderName,usermd,fundingReference.funderName,=
61 fundingReference.funderIdentifier,usermd,fundingReference.
       funderIdentifier,=
62 fundingReference.awardNumber,usermd,fundingReference.awardNumber,=,
63 fundingReference.awardNumber.URI
64 fundingReference.awardTitle
65
66 # Worked
67
68 worked,usermd,worked,=
69
70 # Worked Note
71
72
73 workedNote,usermd,workedNote,=
74
75 # Title
76
77 title,cdl,title,=
78 title.titleType
79 title.lang
80
81
82 # Description
```

```
83
84 description,usermd,description,=
85 description.descriptionType
86 description.lang
87
88 # Resource Type
89
90 resourceType,usermd,resourceType,=,
91 resourceType.resourceTypeGeneral
92
93 # Keywords
94
95 keywords.keyword
96 keywords.keyword.vocabulary
97 keywords.keyword.vocabularyURL
98
99 # Subject
100
101 subjects.subject,usermd,subject,=,
102 subjects.subject.subjectScheme
103 subjects.subject.schemeURI
104 subjects.subject.valueURI
105 subjects.subject.lang
106
107 # Date
108
109 dates.date,usermd,dates.date,=
110 dates.date.dateType
111
112 # Version
113
114
115 version,usermd,version,=
116
117 # Creation Mode
118
119 mode,usermd,mode,=
120
121 # Measured Variables
122
123 measuredVariable.name
124 measuredVariable.symbol
125 measuredVariable.value
126 measuredVariable.encoding
127 measuredVariable.error
128
129 # Controlled Variables
130
```

```
131 controlledVariable.name
132 controlledVariable.symbol
133 controlledVariable.value
134 controlledVariable.encoding
135 controlledVariable.error
136
137 controlledVariable.name,cdl,time:standard_name,=,1
138 controlledVariable.symbol,cdl,double time\(time\),na,1
139 controlledVariable.encoding,cdl,time:unit,=,1
140
141 controlledVariable.name,cdl,tas:standard_name,=,2
142 controlledVariable.value,cdl,tas:_FillValue,=,2
143 controlledVariable.symbol,cdl,float tas,na,2
144 controlledVariable.encoding,cdl,tas:unit,=,2
145
146 controlledVariable.name,usermd,var3.name,=,3
147 controlledVariable.value,mdp,ref_p,=,3
148
149 controlledVariable.name,usermd,var4.name,=,4
150 controlledVariable.value,mdp,pcoupl,=,4
151
152
153
154
155 # Provenance
156
157 provenance.processingStep.type,cdl,experiment_id,=,
158 provenance.processingStep.actor.name,usermd,creator.name,=
159 provenance.processingStep.actor.givenName,usermd,creator.firstname
    ,=
160 provenance.processingStep.actor.familyName,usermd,creator.lastname
    ,=
161 provenance.processingStep.actor.address,usermd,creator.address,=
162 provenance.processingStep.actor.affiliation.name,usermd,creator.
    affiliation.name,=
163 provenance.processingStep.actor.affiliation.address,usermd,creator.
    affiliation.address,=
164 provenance.processingStep.actor.email,usermd,creator.email,=
165 provenance.processingStep.actor.id,usermd,creator.id,=,
166 provenance.processingStep.actor.role,usermd,creator.role,=
167
168
169 provenance.processingStep.date,mdp,At date,:
170
171 provenance.processingStep.method.name.value,usermd,provenance.
    processingStep.method.name.value,=
172 provenance.processingStep.method.name.methodScheme
173 provenance.processingStep.method.name.schemeURI
```

```
174 provenance.processingStep.method.name.valueURI
175 provenance.processingStep.method.description,cdl,comment,=,
176 provenance.processingStep.method.parameter.name
177 provenance.processingStep.method.parameter.symbol
178 provenance.processingStep.method.parameter.value
179 provenance.processingStep.method.parameter.encoding
180 provenance.processingStep.method.parameter.error
181
182 # Error method. Same definition as for the general methods
183 provenance.processingStep.errormethod.name.value
184 provenance.processingStep.errormethod.name.methodScheme
185 provenance.processingStep.errormethod.name.schemeURI
186 provenance.processingStep.errormethod.name.valueURI
187 provenance.processingStep.errormethod.description
188 provenance.processingStep.errormethod.parameter.name
189 provenance.processingStep.errormethod.parameter.symbol
190 provenance.processingStep.errormethod.parameter.value
191 provenance.processingStep.errormethod.parameter.encoding
192 provenance.processingStep.errormethod.parameter.error
193
194
195 # Input file or stream, specified by (P)ID or link and checksum (
      optional)
196 provenance.processingStep.input.id
197 provenance.processingStep.input.id.type
198 provenance.processingStep.input.id.scheme
199 provenance.processingStep.input.link
200 provenance.processingStep.input.checksum
201 provenance.processingStep.input.checksum.type
202
203 provenance.processingStep.input.id,cdl,ozone forcing,na,1
204 provenance.processingStep.input.id,cdl,aerosol optics,na,2
205 provenance.processingStep.input.id,cdl,aerosol MMR,na,3
206 provenance.processingStep.input.id,cdl,carbon scaling,na,4
207 provenance.processingStep.input.id,cdl,GHGs,na,5
208 provenance.processingStep.input.id,cdl,GHG loss rates,na,6
209 provenance.processingStep.input.id,cdl,DMS emissions,na,7
210 provenance.processingStep.input.id,cdl,oxidants,na,8
211 provenance.processingStep.input.id,cdl,SOx emissions,na,9
212
213 # Software or code, that is used for the simulation.
214 provenance.processingStep.tool.name,cdl,source,=,1
215 provenance.processingStep.tool.contributor.name
216 provenance.processingStep.tool.contributor.givenName
217 provenance.processingStep.tool.contributor.familyName
218 provenance.processingStep.tool.contributor.address
219 provenance.processingStep.tool.contributor.affiliation
220 provenance.processingStep.tool.contributor.email
```

```
221 provenance.processingStep.tool.contributor.id
222 provenance.processingStep.tool.contributor.role
223 provenance.processingStep.tool.softwareVersion,log,GROMACS version,
    :
224 provenance.processingStep.tool.operatingSystem,log,Build OS/arch,:
225 provenance.processingStep.tool.programmingLanguage
226 provenance.processingStep.tool.SoftwareSourceCode.id
227 provenance.processingStep.tool.SoftwareSourceCode.id.type
228 provenance.processingStep.tool.SoftwareSourceCode.id.scheme
229 provenance.processingStep.tool.SoftwareSourceCode.link
230 provenance.processingStep.tool.SoftwareSourceCode.checksum
231 provenance.processingStep.tool.SoftwareSourceCode.checksum.type
232 provenance.processingStep.tool.SoftwareApplication.id
233 provenance.processingStep.tool.SoftwareApplication.id.type
234 provenance.processingStep.tool.SoftwareApplication.id.scheme
235 provenance.processingStep.tool.SoftwareApplication.link
236 provenance.processingStep.tool.SoftwareApplication.checksum
237 provenance.processingStep.tool.SoftwareApplication.checksum.type
238 provenance.processingStep.tool.codeRepository
239 provenance.processingStep.tool.license.note
240 provenance.processingStep.tool.license.terms
241 provenance.processingStep.tool.citation
242 provenance.processingStep.tool.referencedPublication.id
243 provenance.processingStep.tool.referencedPublication.id.scheme
244 provenance.processingStep.tool.referencedPublication.id.type
245 provenance.processingStep.tool.referencedPublication.citation,cdl,
    references,=,
246 provenance.processingStep.tool.referencedPublication.url
247
248 # Exact execution command that was called in this processingStep
249
250 provenance.processingStep.executionCommand,cdl,cmd_ln,=,1
251
252
253  # Instrument used in this processingStep
254 provenance.processingStep.instrument.name
255 provenance.processingStep.instrument.description
256 provenance.processingStep.instrument.type
257 provenance.processingStep.instrument.partnum.value
258 provenance.processingStep.instrument.partnum.vendor
259 provenance.processingStep.instrument.serialnum.value
260 provenance.processingStep.instrument.serialnum.vendor
261 provenance.processingStep.instrument.software
262 provenance.processingStep.instrument.software.os
263 provenance.processingStep.instrument.date
264
265 # Output file or stream, specified by (P)ID or link and checksum (
    optional)
```

```
266 provenance.processingStep.output.id
267 provenance.processingStep.output.id.type
268 provenance.processingStep.output.id.scheme
269 provenance.processingStep.output.link
270 provenance.processingStep.output.checksum
271 provenance.processingStep.output.checksum.type
272
273
274
275
276 # Computing environment of this processingStep
277 provenance.processingStep.environment.name
278 provenance.processingStep.environment.compiler.name,log,g++ ,na,1
279 provenance.processingStep.environment.compiler.flags,log,C++
        compiler flags,:,1
280 provenance.processingStep.environment.nodes,job,nodes,=,
281 provenance.processingStep.environment.ppn,job,ppn,=,
282 provenance.processingStep.environment.cpu,log,Build CPU brand,:
283
284 provenance.processingStep.environment.compiler.name,log,gcc ,na,2
285 provenance.processingStep.environment.compiler.flags,log,C compiler
         flags,:,2
286
287
288 provenance.processingStep.order
289
290 # Size
291
292 size
293
294 # System
295
296 system.phase
297 system.phase.component.name
298 system.phase.component.smilesCode
299 system.phase.component.IUPAC
300 system.phase.component.quantity
301 system.phase.component.unit
302 system.phase.component.forcefield.name
303 system.phase.component.forcefield.parameter.name
304 system.phase.component.forcefield.parameter.symbol
305 system.phase.component.forcefield.parameter.value
306 system.phase.component.forcefield.parameter.encoding
307 system.phase.component.forcefield.parameter.error
308 system.component.name
309 system.component.smilesCode
310 system.component.IUPAC
311 system.component.quantity
```

```
312 system.component.unit
313 system.component.forcefield.name
314 system.component.forcefield.parameter.name
315 system.component.forcefield.parameter.symbol
316 system.component.forcefield.parameter.value
317 system.component.forcefield.parameter.encoding
318 system.component.forcefield.parameter.error
319 system.grid.file.id
320 system.grid.file.id.type
321 system.grid.file.id.scheme
322 system.grid.file.link
323 system.grid.file.checksum
324 system.grid.file.checksum.type
325 system.grid.countCells
326 system.grid.countBlocks
327 system.grid.targetCores
328 system.grid.countX
329 system.grid.countY
330 system.grid.countZ
331 system.grid.unit
332 system.grid.distance
333 system.grid.scalingFormula
334 system.grid.point.positionX
335 system.grid.point.positionY
336 system.grid.point.positionZ
337 system.temporalResolution.numberOfTimesteps,mdp,nsteps,=,
338 system.temporalResolution.interval,mdp,dt,=,
339 system.temporalResolution.unit
340 system.temporalResolution.timestep
341
342
343 # Storage information
344 storage.contentLocation.value
345 storage.contentLocation.link
346 storage.storageMedium
347 storage.storageMedium.URI
348
349 # Format information
350 format.formatDesignation.formatName
351 format.formatDesignation.formatVersion
352 format.formatRegistry.name
353 format.formatRegistry.key
354 format.formatRegistry.role
355
356
357 # The PID of the dataset
358 pid
359 pid.type
```

```
360 pid.scheme
361
362 # Checksum of the dataset
363 checksum
364 checksum.algorithm
365
366 # Rights of the dataset
367 rightsStatement.copyrightInformation.note,cdl,acknowledgment,=,
368 rightsStatement.copyrightInformation.status
369 rightsStatement.copyrightInformation.countryCode
370
371
372 #Context objects
373 context.referencePublication.id
374 context.referencePublication.id.type
375 context.referencePublication.id.scheme
376 context.referencePublication.citation
377 context.referencePublication.url
378 context.relatedResource.id
379 context.relatedResource.id.type
380 context.relatedResource.id.scheme
381 context.relatedResource.link
382 context.relatedResource.checksum
383 context.relatedResource.checksum.scheme
384 context.relatedIdentifier
385 context.relatedIdentifier.type
386 context.relatedIdentifier.scheme
```

Listing A.3: Für die Evaluation genutzt Konfigurationsdatei zur Metadatenerfassung von NetCDF-Simulationsdaten.

Anhang B

Appendix: EngMeta XML-Dateien nach Metadatenerfassung

B.1 EngMeta XML für GROMACS-Daten

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<dataset xmlns:ns2="http://www.loc.gov/premis/v3" xmlns:ns3="
    urn:exptml:schema:draft:0.5" xmlns:ns4="
    urn:oasis:names:tc:unitsml:schema:xsd:UnitsMLSchema-1.0">
    <contact>
        <name>Kraus</name>
        <givenName>Hamzeh</givenName>
        <familyName>Kraus</familyName>
        <address>Nobelstr. 19, 70569</address>
        <affiliation>
            <name>Institute of Thermodynamics and Thermal Process
    Engineering, University of Stuttgart</name>
            <address>Nobelstr. 19, 70569</address>
        </affiliation>
        <email>kraus@itt.uni-stuttgart.de</email>
        <id>https://www.itt.uni-stuttgart.de/institut/team/
    mitarbeiter/Kraus-00003/</id>
        <role>ContactPerson</role>
    </contact>
    <creator>
        <name>Hamzeh Kraus</name>
        <givenName>Hamzeh</givenName>
        <familyName>Kraus</familyName>
        <address>Pfaffenwaldring 9, 70569 Stuttgart</address>
        <affiliation>
            <name>ITT</name>
            <address>Pfaffenwaldring 9, 70569 Stuttgart</address>
        </affiliation>
        <email>kraus@itt.uni-stuttgart.de</email>
        <id/>
        <role>Producer</role>
    </creator>
```

```
29      <contributor>
30          <name>Bjoern Schembera</name>
31          <givenName>Bjoern</givenName>
32          <familyName>Schembera</familyName>
33          <affiliation>
34              <name>HLRS</name>
35              <address>Nobelstr. 19, 70569</address>
36          </affiliation>
37          <email>schembera@hlrs.de</email>
38          <id>0000-0003-2860-6621</id>
39          <role>DataManager</role>
40      </contributor>
41      <project level="1">DIPL-ING</project>
42      <fundingReference>
43          <funderName>BMBF</funderName>
44          <funderIdentifier>Other</funderIdentifier>
45          <awardNumber>16FMD008</awardNumber>
46      </fundingReference>
47      <worked>false</worked>
48      <workedNote>After the second run, simulation worked</workedNote
        >
49      <title>The simulation of Hexane</title>
50      <description>Simulation of hexane is done...</description>
51      <resourceType>Dataset</resourceType>
52      <subjects>
53          <subject>Thermodynamics</subject>
54      </subjects>
55      <dates>
56          <date>2019-02-20</date>
57      </dates>
58      <version>1.2</version>
59      <mode>Simulation</mode>
60      <controlledVariable>
61          <name>temperature</name>
62          <value>300</value>
63      </controlledVariable>
64      <controlledVariable>
65          <name>thermostat</name>
66          <value>-1</value>
67      </controlledVariable>
68      <controlledVariable>
69          <name>pressure</name>
70          <value>1.0</value>
71      </controlledVariable>
72      <controlledVariable>
73          <name>barostat</name>
74          <value>isotropic</value>
75      </controlledVariable>
```

```
76      <system>
77          <temporalResolution>
78              <numberOfTimesteps>50000000</numberOfTimesteps>
79              <interval>0.001</interval>
80          </temporalResolution>
81      </system>
82      <provenance>
83          <step>
84              <executionCommand>;    gmx_mpi grompp -f ../_mdp/run.mdp
     -c ../npt/npt.gro -t ../npt/npt.cpt -p ../_top/topol.top -po
    run -o run</executionCommand>
85              <executionCommand>gmx_mpi mdrun -s run.tpr -maxh 24 -
    dlb no -deffnm run -c -e -g -o -cpo</executionCommand>
86              <environment>
87                  <compiler>
88                      <name>C compiler:           /opt/bwhpc/common/
    compiler/gnu/7.1.0/bin/gcc GNU 7.1.0</name>
89                      <flags>-mavx      -O3 -DNDEBUG -funroll-all-
    loops -fexcess-precision=fast</flags>
90                  </compiler>
91              </environment>
92          </step>
93      </provenance>
94      <size>0</size>
95  </dataset>
```

Listing B.1: XML-Datei gemäß EngMeta nach Metadatenerfassung von GROMACS-Simulationsdaten.

B.2 EngMeta XML für EAS3-Daten

```
1  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2  <dataset xmlns:ns2="http://www.loc.gov/premis/v3" xmlns:ns3="
       urn:exptml:schema:draft:0.5" xmlns:ns4="
       urn:oasis:names:tc:unitsml:schema:xsd:UnitsMLSchema-1.0">
3      <contact>
4          <name>Selent</name>
5          <givenName>Bjoern</givenName>
6          <familyName>Selent</familyName>
7          <affiliation>
8              <name>Institut fuer Aerodynamik und Gasdynamik</name>
9              <email>selent@iag.uni-stuttgart.de</email>
10         </affiliation>
11         <email>selent@iag.uni-stuttgart.de</email>
12         <id/>
13     </contact>
14     <contact>
```

```
15          <name>Schembera</name>
16          <givenName>Bjoern</givenName>
17          <familyName>Schembera</familyName>
18          <address>Nobelstr. 19</address>
19          <affiliation>
20              <name>HLRS</name>
21          </affiliation>
22          <email>schembera@hlrs.de</email>
23          <id/>
24          <role>DataManager</role>
25      </contact>
26      <creator>
27          <givenName>Bjoern</givenName>
28          <familyName>Selent</familyName>
29          <affiliation>
30              <name>Institut fuer Aerodynamik und Gasdynamik</name>
31          </affiliation>
32          <email>selent@iag.uni-stuttgart.de</email>
33          <id/>
34      </creator>
35      <contributor>
36          <name>Schembera</name>
37          <givenName>Bjoern</givenName>
38          <familyName>Schembera</familyName>
39          <address>Nobelstr. 19</address>
40          <affiliation>
41              <name>HLRS</name>
42          </affiliation>
43          <email>schembera@hlrs.de</email>
44          <id/>
45          <role>DataManager</role>
46      </contributor>
47      <contributor>
48          <affiliation/>
49          <id/>
50      </contributor>
51      <project level="1">DIPL-ING</project>
52      <fundingReference>
53          <funderName>BMBF</funderName>
54          <awardNumber>16FDM008</awardNumber>
55      </fundingReference>
56      <worked>true</worked>
57      <workedNote>Beim ersten Anlauf...</workedNote>
58      <title>IAG Simulation</title>
59      <description xml:lang="Deutsch">RM-Test with 2D-Baseflow for 3D
        -Case</description>
60      <keywords>
61          <keyword>Aerodynamik</keyword>
```

```
62          <keyword>Stroemung</keyword>
63          <keyword>IAG</keyword>
64          <keyword>HLRS</keyword>
65      </keywords>
66      <subjects>
67          <subject xml:lang="de">Aerodynamik</subject>
68          <subject xml:lang="en">Aerodynamics</subject>
69      </subjects>
70      <dates>
71          <date>4.4.2019</date>
72      </dates>
73      <version>0.67</version>
74      <mode>Simulation</mode>
75      <controlledVariable>
76          <name>Machzahl</name>
77          <value>0.850</value>
78      </controlledVariable>
79      <controlledVariable>
80          <name>Prandtlzahl</name>
81          <value>0.710</value>
82      </controlledVariable>
83      <controlledVariable>
84          <name>Startzeit</name>
85          <value>xxx</value>
86      </controlledVariable>
87      <controlledVariable>
88          <name>Reynoldszahl</name>
89          <value>3000.000</value>
90      </controlledVariable>
91      <system>
92          <grid>
93              <countCells>147456000</countCells>
94              <countX>60</countX>
95              <countY>30</countY>
96              <countZ>128</countZ>
97          </grid>
98          <temporalResolution>
99              <numberOfTimesteps>120000</numberOfTimesteps>
100             <interval>0.00335</interval>
101         </temporalResolution>
102         <boundaryCondition>
103             <parameter>
104                 <name>periodic</name>
105             </parameter>
106         </boundaryCondition>
107     </system>
108     <system>
109         <grid>
```

```
110             <countCells>147456000</countCells>
111             <countX>60</countX>
112             <countY>30</countY>
113             <countZ>128</countZ>
114         </grid>
115         <temporalResolution>
116             <numberOfTimesteps>120000</numberOfTimesteps>
117             <interval>0.00335</interval>
118         </temporalResolution>
119         <boundaryCondition>
120             <parameter>
121                 <name>periodic</name>
122             </parameter>
123         </boundaryCondition>
124     </system>
125     <system>
126         <grid>
127             <countCells>147456000</countCells>
128             <countX>60</countX>
129             <countY>30</countY>
130             <countZ>128</countZ>
131         </grid>
132         <temporalResolution>
133             <numberOfTimesteps>120000</numberOfTimesteps>
134             <interval>0.00335</interval>
135         </temporalResolution>
136         <boundaryCondition>
137             <parameter>
138                 <name>periodic</name>
139             </parameter>
140         </boundaryCondition>
141     </system>
142     <system>
143         <grid>
144             <countCells>147456000</countCells>
145             <countX>60</countX>
146             <countY>30</countY>
147             <countZ>128</countZ>
148         </grid>
149         <temporalResolution>
150             <numberOfTimesteps>120000</numberOfTimesteps>
151             <interval>0.00335</interval>
152         </temporalResolution>
153         <boundaryCondition>
154             <parameter>
155                 <name>periodic</name>
156             </parameter>
157         </boundaryCondition>
```

```
158     </system>
159     <system>
160         <grid>
161             <countCells>147456000</countCells>
162             <countX>60</countX>
163             <countY>30</countY>
164             <countZ>128</countZ>
165         </grid>
166         <temporalResolution>
167             <numberOfTimesteps>120000</numberOfTimesteps>
168             <interval>0.00335</interval>
169         </temporalResolution>
170         <boundaryCondition>
171             <parameter>
172                 <name>periodic</name>
173             </parameter>
174         </boundaryCondition>
175     </system>
176     <system>
177         <grid>
178             <countCells>147456000</countCells>
179             <countX>60</countX>
180             <countY>30</countY>
181             <countZ>128</countZ>
182         </grid>
183         <temporalResolution>
184             <numberOfTimesteps>120000</numberOfTimesteps>
185             <interval>0.00335</interval>
186         </temporalResolution>
187         <boundaryCondition>
188             <parameter>
189                 <name>periodic</name>
190             </parameter>
191         </boundaryCondition>
192     </system>
193     <provenance>
194         <step>
195             <date>00-2-12-28T00:00:00.000+01:00</date>
196             <tool>
197                 <name>ns3dneo</name>
198                 <softwareVersion>0.60</softwareVersion>
199             </tool>
200             <environment>
201                 <compiler>
202                     <name>compiled with Cray Fortran : Version
    8.7.5</name>
```

```
203                     <flags> -O cache2,scalar2,thread2,vector2,mpi0,
    modinline,ipa3,noaggress -O autoprefetch,noautothread,fusion2,
    nomsgs,nonegmsgs,nooverindex -O pattern,shortci
204 rcuit2,unroll2,nozeroinc -h noadd_paren,align_arrays,nobounds,caf,
    noconcurrent,nocontiguous -h nocontiguous_assumed_shape,fma,
    fp_trap,nofunc_trace,noomp_analyze -h noomp_trace,
205 nopat_trace,PIC -h safe_addr,thread_do_concurrent,fp3=approx,
    flex_mp=default          -h alias=default:standard_restrict
        -h dynamic (or -dynamic) -h omp,noacc -h cpu=x
206 86-64,haswell -h network=aries -K trap=fp,divz,inv,ovf -s real64 -
    eh -d abcdefgijnopvzBDEIPQRSZ0 -e mqswACFT -J </flags>
207                 </compiler>
208                 <nodes>160</nodes>
209                 <ppn>24</ppn>
210             </environment>
211         </step>
212     </provenance>
213     <size>90560</size>
214 </dataset>
```

Listing B.2: XML-Datei gemäß EngMeta nach Metadatenerfassung von EAS3-Simulationsdaten.

B.3 EngMeta XML für NetCDF-Daten

```
1  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2  <dataset xmlns:ns2="http://www.loc.gov/premis/v3" xmlns:ns3="
       urn:exptml:schema:draft:0.5" xmlns:ns4="
       urn:oasis:names:tc:unitsml:schema:xsd:UnitsMLSchema-1.0">
3      <contact>
4          <affiliation>
5              <name>"NCAR (National Center for Atmospheric \n",</name
       >
6          </affiliation>
7          <email>"ccsm@ucar.edu" ;</email>
8          <id/>
9      </contact>
10     <contributor>
11         <affiliation/>
12         <id/>
13     </contributor>
14     <project>"IPCC Fourth Assessment" ;</project>
15     <worked>false</worked>
16     <title>"model output prepared for IPCC AR4" ;</title>
17     <subjects/>
18     <dates/>
19     <controlledVariable>
```

```
20          <name>"time" ;</name>
21          <symbol>double time(time) ;</symbol>
22          <encoding>"days since 0000-1-1" ;</encoding>
23      </controlledVariable>
24      <controlledVariable>
25          <name>"air_temperature" ;</name>
26          <symbol>float tas(time, lat, lon) ;</symbol>
27          <value>1.e+20f ;</value>
28          <encoding>"K" ;</encoding>
29      </controlledVariable>
30
31      <provenance>
32          <step>
33              <type>"720 ppm stabilization experiment (SRESA1B)" ;</
        type>
34              <method>
35                  <description>"This simulation was initiated from
        year 2000 of \n",</description>
36                  <parameter/>
37              </method>
38              <input>
39                  <id>"SOx emissions     :
        SOx_emissions_A1B_128x256_L2_1990-2100_c040608.nc\n",</id>
40              </input>
41              <input>
42                  <id>"aerosol optics    : AerosolOptics_c040105.nc\n"
        ,</id>
43              </input>
44              <input>
45                  <id>"aerosol MMR       :
        AerosolMass_V_128x256_clim_c031022.nc\n",</id>
46              </input>
47              <input>
48                  <id>"carbon scaling    : carbonscaling_A1B_1990-2100
        _c040609.nc\n",</id>
49              </input>
50              <input>
51                  <id>"GHGs              : ghg_ipcc_A1B_1870-2100
        _c040521.nc\n",</id>
52              </input>
53              <input>
54                  <id>"GHG loss rates    : noaamisc.r8.nc\n",</id>
55              </input>
56              <input>
57                  <id>"DMS emissions     :
        DMS_emissions_128x256_clim_c040122.nc\n",</id>
58              </input>
59              <input>
```

```
60                  <id>"oxidants          :
    oxid_128x256_L26_clim_c040112.nc\n",</id>
61              </input>
62              <tool>
63                  <name>"CCSM3.0, version beta19 (2004): \n",</name>
64                  <referencePublication>
65                      <id/>
66                      <citation>"Collins, W.D., et al., 2005:\n",</
    citation>
67                  </referencePublication>
68              </tool>
69              <executionCommand>"bds -x 256 -y 128 -m 23 -o /data/
    zender/data/dst_T85.nc" ;</executionCommand>
70          </step>
71      </provenance>
72      <size>11882</size>
73      <rightsStatement>
74          <ns2:copyrightInformation>
75              <ns2:copyrightNote>" Any use of CCSM data should
    acknowledge the contribution\n",</ns2:copyrightNote>
76          </ns2:copyrightInformation>
77      </rightsStatement>
78  </dataset>
```

Listing B.3: XML-Datei gemäß EngMeta nach Metadatenerfassung von NetCDF-Simulationsdaten.

Abkürzungsverzeichnis

AIP	**A**rchival **I**nformation **P**ackage
API	**A**pplication **P**rogramming **I**nterface
ARC	**A**ustralian **R**esearch **C**ouncil
BMBF	**B**undes**m**inisterium für **B**ildung und **F**orschung
CCSDS	**C**onsultative **C**ommittee for **S**pace **D**ata **S**ystems
CDO	**C**hief **D**ata **O**fficer
CERA	**C**limate and **E**nvironmental **R**etrival and **A**archive
CF	**C**limate and **F**orecast Metadata
CFD	**C**omputational **F**luid **D**ynamics, Numerische Strömungsmechanik
CIO	**C**hief **I**nformation **O**fficer
CLE	**C**ray **L**inux **E**nvironment
CML	**C**hemical **M**arkup **L**anguage
CMIP	**C**oupled **M**odel **I**ntercomparison **P**roject
CPU	**C**entral **P**rocessing **U**nit, Hauptprozessor
CTO	**C**hief **T**echnology **O**fficer
DD	**D**unkle **D**aten
DaRUS	**Da**ten**r**epositorium der **U**niversität **S**tuttgart
DCMI	**D**ublin **C**ore **M**etadata **I**nitiative
DFG	**D**eutsche **F**orschungs**g**emeinschaft
DGSVO	**D**aten**s**schutz-**G**rund**ver**o**r**dnung
DIP	**D**issemination **I**nformation **P**ackage
DIPL-ING	**D**atenmanagement in **I**nfrastrukturen, **P**rozessen und **L**ebenszyklen in den **Ing**enieurwissenschaften
DKRZ	**D**eutsches **K**lima**r**echen**z**entrum
DMP	**D**aten**m**anagement**p**lan
DOI	**D**igital **O**bject **I**dentifier
DWD	**D**eutscher **W**etter**d**ienst
EAS3	**E**in-**A**usgabe-System **3**
EMMC	**E**uropean **M**aterials **M**odelling **C**ouncil
EngMeta	**Eng**ineering **Meta**data
EU	**E**uropäische **U**nion
ESGF	**E**arth **S**ystem **G**rid **F**ederation
ExptML	**Exp**erimen**t** **M**arkup **L**anguage
FA	**F**unktionale **A**nforderung
FAIR	**F**indable, **A**ccessible, **I**nteroperable, **R**e-usable

FDM	**F**orschungs**d**aten**m**anagement
GROMACS	**Gro**ningen **M**achine for **C**hemical **S**imulations
HLRS	**H**öchst**L**eistungs**R**echenzentrum **S**tuttgart
HPC	**H**igh-**P**erformance **C**omputing / Höchstleistungsrechnen
HPSS	**H**igh **P**erformance **S**torage **S**ystem
HSM	**H**ierarchical **S**torage **M**anagement
IAG	**I**nstitut für **A**ero- und **G**asdynamik
ISO	Internationale Organisation für Normung
IEEE	**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers
IPCC	**I**ntergovernmental **P**anel on **C**limate **C**hange
ITT	**I**nstitut für Technische **T**hermodynamik und **T**hermische Verfahrenstechnik
JAXB	**J**ava **A**rchitecture for **X**ML **B**inding
JSON	**J**ava**S**cript **O**bject **N**otation
MASi	**M**etadata Management for **A**pplied **Sci**ences
MD	**M**eta**d**aten
MoSGrid	**Mo**lecular **S**imulation **Grid**
MSML	**M**olecular **M**arkup **S**imulation **L**anguage
MTSR	International Conference on **Met**adata and **S**emantics **R**esearch
NFA	**N**icht**f**unktionale **A**nforderung
NetCDF	**Net**work **C**ommon **D**ata **F**ormat
NSF	**N**ational **S**cience **F**oundation
OAIS	**O**pen **A**rchival **I**nformation **S**ystem
OECD	**O**rganisation for **E**conomic **C**o-operation and **D**evelopment
PaaS	**P**latform **a**s **a** **S**ervice
PB	**P**eta**B**ytes
PBS	**P**ortable **B**atch **S**system
PID	**P**ersistent **Id**entifier / Persistente Kennung
POSIX	**P**ortable **O**perating **S**ystem **I**nterface
PREMIS	**Pre**servation **M**etadata: **I**mplementation **S**trategies
RAID	**R**edundant **A**rray of **I**ndependent **D**isks
RHEL	**R**ed **H**at **E**nterprise **L**inux
RDD	**R**esilient **D**istributed **D**ataset
SaaS	**S**oftware **a**s **a** **S**ervice
SDO	**S**cientific **D**ata **O**fficer
SIP	**S**ubmission **I**nformation **P**ackage
SSD	**S**olid-**s**ate **D**rive
SFB	**S**onder**f**orschungs**b**ereich
TB	**T**era**b**ytes
TCO	**T**otal **C**ost of **O**wnership, Gesamtkosten des Betriebs
TIK	**T**echnische **I**nformations- und **K**ommunikationsdienste der Universität Stuttgart
UB	**U**niversitäts**b**ibliothek
W3C	**W**orld Wide Web **C**onsortium

WCRP	**W**orld **C**limate **R**esearch **P**rogramme
WDCC	**W**orld **D**ata **C**entre for **C**limate
XML	E**X**tensible **M**arkup **L**anguage
XSD	**X**ML **S**chema **D**efinition

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Algorithmen

Listings

Literatur

Adorf, Carl S u. a. (2018). „Simple data and workflow management with the signac framework“. In: *Computational Materials Science* 146, S. 220–229. DOI: 10.1016/j.commatsci.2018.01.035.

Allcock, Bill u. a. (2002). „Data management and transfer in high-performance computational grid environments“. In: *Parallel Computing* 28.5, S. 749 –771. ISSN: 0167-8191. DOI: https://doi.org/10.1016/S0167-8191(02)00094-7. URL: http://www.sciencedirect.com/science/article/pii/S0167819102000947.

Ammann, Noémie u. a. (2011). *DataCite metadata schema for the publication and citation of research data*. Zugegriffen: 27.4.2019. DOI: 10.5438/0010. URL: https://schema.datacite.org/meta/kernel-3.1/index.html.

Anderson, Chris (2004). *The Long Tail*. https://www.wired.com/2004/10/tail/. Zugegriffen: 25.4.2019.

Arora, Ritu (2015). „Data Management: State-of-the-Practice at Open-Science Data Centers“. In: *Handbook on Data Centers*. Hrsg. von Samee U. Khan und Albert Y. Zomaya. Springer New York, S. 1095–1108. ISBN: 978-1-4939-2092-1. DOI: 10.1007/978-1-4939-2092-1_37. URL: http://dx.doi.org/10.1007/978-1-4939-2092-1_37.

Askhoj, Jan, Shigeo Sugimoto und Mitsuharu Nagamori (2011). „Preserving records in the cloud“. In: *Records Management Journal* 21.3, S. 175–187. DOI: 10.1108/09565691111186858. URL: https://doi.org/10.1108/09565691111186858.

Barisits, Martin (2018). *The data management system RucioEvolution for LHC Run-3 and beyond ATLAS*. https://cds.cern.ch/record/2625457/files/ATL-SOFT-SLIDE-2018-401.pdf. Zugegriffen: 3.5.2019.

Barisits, Martin-Stefan u. a. (2017). *The ATLAS Data Management System Rucio: Supporting LHC Run-2 and beyond*. Techn. Ber. ATL-SOFT-PROC-2017-064. 3. Geneva: CERN. URL: https://cds.cern.ch/record/2291104.

Bretherton, Francis P und Paul T Singley (1994). „Metadata: A user's view“. In: *Seventh International Working Conference on Scientific and Statistical Database Management*. IEEE, S. 166–174. DOI: 10.1109/SSDM.1994.336950.

Bruce, Thomas R und D Hillmann (2004). „Metadata in Practice“. In: *The continuum of metadata quality: defining, expressing, exploiting*, S. 238–256. URL: https://hdl.handle.net/1813/7895.

Bungartz, Hans-Joachim u. a. (2013). „Molekulardynamik“. In: *Modellbildung und Simulation*. Berlin Heidelberg: Springer, S. 305–323. ISBN: 978-3-642-37655-9. DOI: 10.1007/978-3-642-37656-6.

Büttner, Stephan, Hans-Christoph Hobohm und Lars Müller (2011). *Handbuch Forschungsdatenmanagement*. Bad Honnef: Bock und Herchen. ISBN: 978-3-88347-283-6.

Cafarella, M. u. a. (2016). „Dark Data: Are we solving the right problems?“ In: *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, S. 1444–1445. DOI: 10.1109/ICDE.2016.7498366.

Calhoun, S. Patrick u. a. (2016). „Large scale research data archiving: Training for an inconvenient technology“. In: *Journal of Computational Science*. ISSN: 1877-7503. DOI: https://doi.org/10.1016/j.jocs.2016.07.005. URL: http://www.sciencedirect.com/science/article/pii/S1877750316301132.

Caplan, Priscilla (2009). *Understanding PREMIS*. https://www.loc.gov/standards/premis/understanding-premis-rev2017.pdf. Zugegriffen: 28.4.2019. Library of Congress Washington DC, USA.

Chiang, Gen-Tao u. a. (2011). „Implementing a genomic data management system using iRODS in the Wellcome Trust Sanger Institute“. In: *BMC Bioinformatics* 12.1, S. 361. ISSN: 1471-2105. DOI: 10.1186/1471-2105-12-361. URL: https://doi.org/10.1186/1471-2105-12-361.

CISCO Systems (2016). *Cisco global cloud index: Forecast and methodology 2015–2020*. https://www.cisco.com/c/dam/m/en_us/service-provider/ciscoknowledgenetwork/files/622_11_15-16-Cisco_GCI_CKN_2015-2020_AMER_EMEAR_NOV2016.pdf. Zugegriffen: 28.4.2019.

Courtland, R. (2016). „Transistors could stop shrinking in 2021“. In: *IEEE Spectrum* 53.9, S. 9–11. ISSN: 0018-9235. DOI: 10.1109/MSPEC.2016.7551335.

Cox, Andrew M. und Stephen Pinfield (2014). „Research Data Management and Libraries: Current Cctivities and Future Priorities“. In: *Journal of Librarianship and Information Science* 46.4, S. 299–316. DOI: 10.1177/0961000613492542. URL: http://dx.doi.org/10.1177/0961000613492542.

Deelman, Ewa u. a. (2010). „Metadata and provenance management“. In: *arXiv preprint arXiv:1005.2643*. URL: https://arxiv.org/pdf/1005.2643.pdf.

DFG (2013). *Safeguarding Good Scientific Practice*. http://www.dfg.de/download/pdf/dfg_im_profil/reden_stellungnahmen/download/empfehlung_wiss_praxis_1310.pdf. Zugegriffen: 28.4.2019.

— (2018). *Proposal Preparation Instructions*. https://www.dfg.de/formulare/54_01/54_01_de.pdf. Zugegriffen: 28.4.2019.

Edwards, Paul N. u. a. (2011). „Science friction: Data, metadata, and collaboration“. In: *Social Studies of Science* 41.5. PMID: 22164720, S. 667–690. DOI: 10.1177/0306312711413314. eprint: https://doi.org/10.1177/0306312711413314. URL: https://doi.org/10.1177/0306312711413314.

EU (2016a). *European Cloud Initiative - Building a competitive data and knowledge economy in Europe*. http://ec.europa.eu/newsroom/dae/document.cfm?doc_id=15266. Zugegriffen: 28.4.2019.

— (2016b). *H2020 Programme Guidelines on FAIR Data Management in Horizon 2020*. http://ec.europa.eu/research/participants/data/ref/h2020/grants_manual/hi/oa_pilot/h2020-hi-oa-data-mgt_en.pdf. Zugegriffen: 28.4.2019.

— (2018). *Turning FAIR into Reality*. https://ec.europa.eu/info/sites/info/files/turning_fair_into_reality_1.pdf. Zugegriffen: 28.4.2019. DOI: 10.2777/1524.

Eyring, V. u. a. (2016). „Overview of the Coupled Model Intercomparison Project Phase 6 (CMIP6) experimental design and organization“. In: *Geoscientific Model Development* 9.5, S. 1937–1958. DOI: 10.5194/gmd-9-1937-2016. URL: https://www.geosci-model-dev.net/9/1937/2016/.

Faulhaber, P. (2015). *Investing in the Future of Tape Technology*. https://indico.bnl.gov/event/1955/contributions/4055/attachments/3629/4283/0829_0900_PF_HUF_Meeting_Aug_2016_PV.pdf. Presentation, HPSS User Forum, New York City.

Fehr, Jörg u. a. (2016). „Best practices for replicability, reproducibility and reusability of computer-based experiments exemplified by model reduction software“. In: *arXiv preprint arXiv:1607.01191*. URL: https://arxiv.org/pdf/1607.01191.pdf.

Foster, Ian und Carl Kesselman, Hrsg. (1999). *The Grid: Blueprint for a New Computing Infrastructure*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN: 1-55860-475-8.

Gallas, E J u. a. (2012). „Conditions and configuration metadata for the ATLAS experiment“. In: *Journal of Physics: Conference Series* 396.5, S. 052033. DOI: 10.1088/1742-6596/396/5/052033. URL: https://doi.org/10.1088%2F1742-6596%2F396%2F5%2F052033.

Gavrilis, Dimitris u. a. (2015). „Measuring Quality in Metadata Repositories“. In: *Research and Advanced Technology for Digital Libraries*. Hrsg. von Sarantos Kapidakis, Cezary Mazurek und Marcin Werla. Cham: Springer International Publishing, S. 56–67. ISBN: 978-3-319-24592-8. DOI: 10.1007/978-3-319-24592-8_5.

Gesing, S. u. a. (2015). „Science gateways - leveraging modeling and simulations in HPC infrastructures via increased usability“. In: *2015 International Conference on High Performance Computing Simulation (HPCS)*, S. 19–26. DOI: 10.1109/HPCSim.2015.7237017.

Goetz, T. (2007). „Freeing the dark data of failed scientific experiment“. In: *Wired Magazine* 15.10, S. 7–12. URL: http://www.wired.com/science/discoveries/magazine/15-10/st_essay.

Gray, Jim u. a. (2005). „Scientific Data Management in the Coming Decade“. In: *SIGMOD Rec.* 34.4, S. 34–41. ISSN: 0163-5808. DOI: 10.1145/1107499.1107503. URL: http://doi.acm.org/10.1145/1107499.1107503.

Greenberg, Jane (2003). „Metadata and the world wide web“. In: *Encyclopedia of library and information science* 3, S. 1876–1888.

— (2004). „Metadata Extraction and Harvesting“. In: *Journal of Internet Cataloging* 6.4, S. 59–82. DOI: 10.1300/J141v06n04_05. eprint: https://doi.org/10.1300/J141v06n04_05. URL: https://doi.org/10.1300/J141v06n04_05.

Grunzke, Richard (2016). „Generic Metadata Handling in Scientific Data Life Cycles“. Diss. Technische Universität Dresden.

Grunzke, Richard u. a. (2014). „Standards-based metadata management for molecular simulations“. In: *Concurrency and Computation: Practice and Experience* 26.10, S. 1744–1759. DOI: 10.1002/cpe.3116. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.3116. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.3116.

Grunzke, Richard u. a. (2019). „The MASi repository service — Comprehensive, metadata-driven and multi-community research data management“. In: *Future Generation Computer Systems* 94, S. 879 –894. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2017.12.023. URL: http://www.sciencedirect.com/science/article/pii/S0167739X17305344.

Hanushevsky, Andrew u. a. (2018). „Xcache in the ATLAS Distributed Computing Environment“. In: URL: http://cds.cern.ch/record/2625240.

Hartmann, Stephan (1996). „The World as a Process“. In: *Modelling and Simulation in the Social Sciences from the Philosophy of Science Point of View*. Hrsg. von Rainer Hegselmann, Ulrich Mueller und Klaus G. Troitzsch. Dordrecht: Springer Netherlands, S. 77–100. ISBN: 978-94-015-8686-3. DOI: 10.1007/978-94-015-8686-3_5. URL: https://doi.org/10.1007/978-94-015-8686-3_5.

Hasse, Hans und Johannes Lenhard (2017). „Boon and Bane: On the Role of Adjustable Parameters in Simulation Models“. In: *Mathematics as a Tool: Tracing New Roles of Mathematics in the Sciences*. Hrsg. von Johannes Lenhard und Martin Carrier. Cham: Springer International Publishing, S. 93–115. ISBN: 978-3-319-54469-4. DOI: 10.1007/978-3-319-54469-4_6. URL: https://doi.org/10.1007/978-3-319-54469-4_6.

Heene, M. u. a. (2016). „Automatic Metadata Generation for Dark Data to Support Information Systems“. In: *AGU Fall Meeting Abstracts*. URL: http://adsabs.harvard.edu/abs/2016AGUFMPA13A1971H.

Heery, Rachel und Manjula Patel (2000). „Application Profiles: Mixing and Matching Metadata Schemas“. In: *Ariadne* 25. ISSN: 1361-3200. URL: http://www.ariadne.ac.uk/issue/25/app-profiles/.

Heidorn, P. Bryan (2008). „Shedding Light on the Dark Data in the Long Tail of Science“. In: *Library Trends* 57.2, S. 280–299. DOI: 10.1353/lib.0.0036.

Heidorn, P. Bryan, Gretchen R Stahlman und Julie Steffen (2018). „Astrolabe: Curating, Linking, and Computing Astronomy's Dark Data“. In: *The Astrophysical Journal Supplement Series* 236.1, S. 3. DOI: 10.3847/1538-4365/aab77e.

Hess, Berk u. a. (2013). „GROMACS 4.5: a high-throughput and highly parallel open source molecular simulation toolkit“. In: *Bioinformatics* 29.7, S. 845–854. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btt055. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/29/7/845/17343875/btt055.pdf. URL: https://doi.org/10.1093/bioinformatics/btt055.

Hey, Anthony JG und Anne E Trefethen (2003). „The data deluge: An e-science perspective“. In: S. 809–824. URL: https://eprints.soton.ac.uk/257648/1/The_Data_Deluge.pdf.

Hey, Tony, Stewart Tansley und Kristin Tolle (2009). *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research. ISBN: 978-0-9825442-0-4. URL: https://www.microsoft.com/en-us/research/publication/fourth-paradigm-data-intensive-scientific-discovery/.

Hick, Jason (2010). *HPSS in the Extreme Scale Era: Report to DOE Office of Science on HPSS in 2018-2022*. Techn. Ber. LBNL-3877E. URL: https://escholarship.org/uc/item/4wn1s2d3.

— (2013). *The Fifth Workshop on HPC Best Practices: File Systems and Archives*. Techn. Ber. LBNL-5262E. URL: https://escholarship.org/uc/item/35z307bk.

Iglezakis, Dorothea und Björn Schembera (2018). „Anforderungen der Ingenieurwissenschaften an das Forschungsdatenmanagement der Universität Stuttgart-Ergebnisse der Bedarfsanalyse des Projektes DIPL-ING“. In: *o-bib. Das offene Bibliotheksjournal/Herausgeber VDB* 5.3, S. 46–60. DOI: 10.5282/o-bib/2018H3S46-60.

Information and documentation – Digital object identifier system (2012). Bd. 2012. ISO, Geneva, Switzerland.

Jensen, Uwe (2011). „Datenmanagementpläne“. In: *Handbuch Forschungsdatenmanagement*. Bad Honnef: Bock u. Herchen, S. 71–82.

Jones, Stephanie N. u. a. (2011). „Easing the Burdens of HPC File Management“. In: *Proceedings of the Sixth Workshop on Parallel Data Storage*. PDSW '11. Seattle, Washington, USA: ACM, S. 25–30. ISBN: 978-1-4503-1103-8. DOI: 10.1145/2159352.2159359. URL: http://doi.acm.org/10.1145/2159352.2159359.

Lauber-Rönsberg, Anne, Philipp Krahn und Paul Baumann (2018). *Gutachten zu den rechtlichen Rahmenbedinungen des Forschungsdatenmanagements*. https://tudata-test.ulb.tu-darmstadt.de/bitstream/handle/123456789/863/Gutachten_rechtl_Rahmenbedingungen_FDM_KURZFASSUNG.pdf?sequence=1.

Lautenschlager, Michael (2011). „Institutionalisierte "Data Curation Services"“. In: *Handbuch Forschungsdatenmanagement*. Bad Honnef: Bock u. Herchen, S. 149–156.

Lautenschlager, Michael u. a. (1998). *The CERA-2 data model*. Zugegriffen: 28.4.2019. URL: https://www.pik-potsdam.de/cera/Descriptions/Publications/Papers/9807_DKRZ_TechRep15/cera2.pdf.

Lee, Yang u. a. (2014). „A cubic framework for the chief data officer: Succeeding in a world of big data“. In: Letzter Zugriff 25.2.2019. DOI: 1721.1/103027. URL: https://dspace.mit.edu/bitstream/handle/1721.1/103027/esd-wp-2014-34.pdf?sequence=1.

Liang, Shuo u. a. (2015). „iCurate: A Research Data Management System“. In: *Multidisciplinary Trends in Artificial Intelligence: 9th International Workshop, MIWAI 2015, Fuzhou, China, November 13-15, 2015, Proceedings*. Hrsg. von Antonis Bikakis und Xianghan Zheng. Cham: Springer International Publishing, S. 39–47. ISBN: 978-3-319-26181-2. DOI: 10.1007/978-3-319-26181-2_4. URL: http://dx.doi.org/10.1007/978-3-319-26181-2_4.

Ludewig, Jochen und Horst Lichter (2013). *Software Engineering: Grundlagen, Menschen, Prozesse, Techniken*. Heidelberg: dpunkt. verlag.

Ludwig, Jens und Harry Enke (2013). *Leitfaden zum Forschungsdaten-Management. Handreichungen aus dem WissGrid-Projekt*. Glückstadt: Verlag Werner Hülsbusch.

Ludwig, Thomas und Beate Geyer (2019). „Reproduzierbarkeit“. In: *Informatik Spektrum* 42.1, S. 48–52. ISSN: 1432-122X. DOI: 10.1007/s00287-019-01149-2. URL: https://doi.org/10.1007/s00287-019-01149-2.

Mattmann, Chris A (2013). „Computing: A vision for data science“. In: *Nature* 493.7433, S. 473–475. DOI: 10.1038/493473a. URL: http://dx.doi.org/10.1038/493473a.

Michener, William K. u. a. (1997). „Nongeospatial Metadata for the Ecological Sciences“. In: *Ecological Applications* 7.1, S. 330–342. DOI: 10.2307/2269427. URL: https://www.jstor.org/stable/2269427.

Murray-Rust, Peter und Henry S. Rzepa (2011). „CML: Evolution and design“. In: *Journal of Cheminformatics* 3.1, S. 44. ISSN: 1758-2946. DOI: 10.1186/1758-2946-3-44. URL: https://doi.org/10.1186/1758-2946-3-44.

Neumann, Janna und Jan Brase (2014). „DataCite and DOI names for research data“. In: *Journal of Computer-Aided Molecular Design* 28.10, S. 1035–1041. ISSN: 1573-4951. DOI: 10.1007/s10822-014-9776-5. URL: https://doi.org/10.1007/s10822-014-9776-5.

NISO (2007). *A framework of guidance for building good digital collections*. https://www.niso.org/sites/default/files/2017-08/framework3.pdf.

NSF (2014). *Grant Proposal Guide Chapter II.C.2.j*. https://www.nsf.gov/pubs/policydocs/pappguide/nsf15001/gpg_2.jsp#dmp.

OAIS (2012). *Reference Model for an Open Archival Information System*. Techn. Ber. 650.0-M-2 (Magenta Book) Issue 2. CCSDS.

Odebrecht, Carolin (2018). „MKM – ein Metamodell für Korpusmetadaten“. Diss. Humboldt-Universität zu Berlin.

Ören, TI (2002). „Rationale for a code of professional ethics for simulationists“. In: *Summer Computer Simulation Conference*. Society for Computer Simulation International; 1998, S. 428–433.

Pappenberger, Karlheinz (2016). „bwFDM-Communities–Wissenschaftliches Datenmanagement an den Universitäten Baden-Württembergs“. In: *Bibliothek Forschung und Praxis* 40.1, S. 21–25.

Park, Jung-ran und Andrew Brenza (2015). „Evaluation of Semi-Automatic Metadata Generation Tools: A Survey of the Current State of the Art“. In: *Information Technology and Libraries* 34.3, S. 22–42. DOI: 10.6017/ital.v34i3.5889. URL: https://ejournals.bc.edu/index.php/ital/article/view/5889.

Parker-Wood, Aleatha u. a. (2013). „Examining Extended and Scientific Metadata for Scalable Index Designs“. In: *Proceedings of the 6th International Systems and Storage Conference*. SYSTOR '13. Haifa, Israel: ACM, 4:1–4:6. ISBN: 978-1-4503-2116-7. DOI: 10.1145/2485732.2485754. URL: http://doi.acm.org/10.1145/2485732.2485754.

Petersen, Alexander Michael u. a. (2014). „Reputation and impact in academic careers“. In: *Proceedings of the National Academy of Sciences* 111.43, S. 15316–15321. ISSN: 0027-8424. DOI: 10.1073/pnas.1323111111. eprint: https://www.pnas.org/content/111/43/15316.full.pdf. URL: https://www.pnas.org/content/111/43/15316.

Phadungsukanan, Weerapong u. a. (2012). „The semantics of Chemical Markup Language (CML) for computational chemistry : CompChem“. In: *Journal of Cheminformatics* 4.1, S. 15. ISSN: 1758-2946. DOI: 10.1186/1758-2946-4-15. URL: https://doi.org/10.1186/1758-2946-4-15.

Potthoff, Jan u. a. (2014). „Anforderungen eines nachhaltigen, disziplinübergreifenden Forschungsdaten-Repositoriums“. In: *7. DFN-Forum - Kommunikationstechnologien*. Hrsg. von Paul Müller u. a. Bonn: Gesellschaft für Informatik e.V., S. 44136.

Prabhune, Ajinkya (2018). „Generic and Adaptive Metadata Management Framework for Scientific Data Repositories“. Diss. Universität Heidelberg. DOI: 10.11588/heidok.00024044.

PREMIS (2015). *The PREMIS Schema Definition*. http://www.loc.gov/standards/premis/premis.xsd.

Purss, Matthew B.J. u. a. (2015). „Unlocking the Australian Landsat Archive – From dark data to High Performance Data infrastructures“. In: *GeoResJ* 6. Rescuing Legacy Data for Future Science, S. 135 –140. ISSN: 2214-2428. DOI: https://doi.org/10.1016/j.grj.2015.02.010. URL: http://www.sciencedirect.com/science/article/pii/S2214242815000182.

Quantum White Paper. LTO: The New "Enterprise Tape Drive" (2018). Zugegriffen: 28.4.2019. URL: http://www.quantum.com/iqdoc/doc.aspx?id=15146.

Rat für Informationsinfrastrukturen (2016). *Begriffsklärungen: Bericht des Redaktionsausschusses Begriffe an den RfII*. URL: http://www.rfii.de/download/rfii-berichte-no-1/.

Razum, Matthias (2011). „Systeme und Systemarchitekturen für das Datenmanagement“. In: *Handbuch Forschungsdatenmanagement*. Bad Honnef: Bock u. Herchen, S. 123–138.

Reilly, Susan u. a. (2011). *Report on integration of data and publications*. Zugrgriffen: 3.5.2019. URL: http://www.alliancepermanentaccess.org/wp-content/uploads/downloads/2011/11/ODE-ReportOnIntegrationOfDataAndPublications-1\%5f1.pdf.

Rekawek, Tomasz, Piotr Bała und Krzysztof Benedyczak (2011). „Distributed Storage Management Service in UNICORE“. In: *Schriften des Forschungszentrums Jülich IAS Series Volume 9*, S. 125.

Resch, Michael und Andreas Kaminski (2019). „The Epistemic Importance of Technology in Computer Simulation and Machine Learning“. In: *Minds and Machines*. ISSN: 1572-8641. DOI: 10.1007/s11023-019-09496-5. URL: https://doi.org/10.1007/s11023-019-09496-5.

Roberts, Malcolm J u. a. (2015). „Tropical cyclones in the UPSCALE ensemble of high-resolution global climate models“. In: *Journal of Climate* 28.2, S. 574–596. DOI: 10.1175/JCLI-D-14-00131.1.

Schappals, Michael u. a. (2017). „Round Robin Study: Molecular Simulation of Thermodynamic Properties from Models with Internal Degrees of Freedom“. In: *Journal of Chemical Theory and Computation* 13.9. PMID: 28738147, S. 4270–4280. DOI: 10.1021/acs.jctc.7b00489. eprint: https://doi.org/10.1021/acs.jctc.7b00489. URL: https://doi.org/10.1021/acs.jctc.7b00489.

Schembera, Björn und Thomas Bönisch (2017). „Challenges of Research Data Management for High Performance Computing“. In: *Research and Advanced Technology for Digital Libraries*. Hrsg. von Jaap Kamps u. a. Cham: Springer International Publishing, S. 140–151. ISBN: 978-3-319-67008-9.

Schembera, Björn und Juan M. Durán (2019). „Dark Data as the New Challenge for Big Data Science and the Introduction of the Scientific Data Officer“. In: *Philosophy & Technology*. ISSN: 2210-5441. DOI: 10.1007/s13347-019-00346-x. URL: https://doi.org/10.1007/s13347-019-00346-x.

Schembera, Björn und Dorothea Iglezakis (2019). „The Genesis of EngMeta - A Metadata Model for Research Data in Computational Engineering“. In: *Metadata and Semantic Research*. Hrsg. von Emmanouel Garoufallou u. a. Cham: Springer International Publishing, S. 127–132. ISBN: 978-3-030-14401-2.

Spindler, Gerald und Tobias Hillegeist (2011). „Rechtliche Probleme der elektronischen Langzeitarchivierung von Forschungsdaten“. In: *Handbuch Forschungsdatenmanagement*. Bad Honnef: Bock u. Herchen, S. 63–70.

Stephan, Simon u. a. (2019). „MolMod – an open access database of force fields for molecular simulations of fluids“. In: *Molecular Simulation* 0.0, S. 1–9. DOI: 10.1080/08927022.2019.1601191. eprint: https://doi.org/10.1080/08927022.2019.1601191. URL: https://doi.org/10.1080/08927022.2019.1601191.

Stockhause, Martina und Michael Lautenschlager (2017). „CMIP6 data citation of evolving data“. In: *Data Science Journal* 16.

Stodden, Victoria (2018). *Assessing Reproducibility: An Astrophysical Example of Computational Uncertainty in the HPC Context*. http://web.stanford.edu/~vcs/talks/ResCuESC2018-STODDEN.pdf.

Tani, Alice, Leonardo Candela und Donatella Castelli (2013). „Dealing with metadata quality: The legacy of digital library efforts“. In: *Information Processing & Management* 49.6, S. 1194 –1205. ISSN: 0306-4573. DOI: https://doi.org/10.1016/j.ipm.2013.05.003. URL: http://www.sciencedirect.com/science/article/pii/S0306457313000526.

Trajanov, Dimitar u. a. (2018). „Dark data in internet of things (iot): Challenges and opportunities“. In: *7th Small Systems Simulation Symposium*, S. 1–8.

Tristam, Frank und Achim Streit (2015). *Öffentlicher Abschlussbericht von bwFDM-Communities*. https://bwfdm.scc.kit.edu/downloads/Abschlussbericht.pdf.

Whitten, Dwayne (2008). „The Chief Information Security Officer: An Analysis of the Skills Required for Success“. In: *Journal of Computer Information Systems* 48.3, S. 15–19. DOI: 10.1080/08874417.2008.11646017. eprint: https://www.tandfonline.com/doi/pdf/10.1080/08874417.2008.11646017. URL: https://www.tandfonline.com/doi/abs/10.1080/08874417.2008.11646017.

Wienke, Sandra u. a. (2015). „Modeling the Productivity of HPC Systems on a Computing Center Scale“. In: *High Performance Computing*. Hrsg. von Julian M. Kunkel und Thomas Ludwig. Cham: Springer International Publishing, S. 358–375. ISBN: 978-3-319-20119-1.

Wilkinson, Mark D. u. a. (2016). „The FAIR Guiding Principles for scientific data management and stewardship“. In: *Scientific Data* 3, S. 160018.

Wilms, Konstantin u. a. (2018). „Do researchers dream of research data management?“ In: *Proceedings of the 51st Hawaii International Conference on System Sciences*. DOI: 10125/50445.

Winsberg, Eric (2003). „Simulated Experiments: Methodology for a Virtual World“. In: *Philosophy of Science* 70.1, S. 105–125. DOI: 10.1086/367872. eprint: https://doi.org/10.1086/367872. URL: https://doi.org/10.1086/367872.

— (2009). „A tale of two methods“. In: *Synthese* 169.3, S. 575–592. ISSN: 1573-0964. DOI: 10.1007/s11229-008-9437-0. URL: https://doi.org/10.1007/s11229-008-9437-0.

Yang, Wei u. a. (2018). *Xcache in ATLAS Distributed Computing*. Techn. Ber. ATL-SOFT-PROC-2018-031. Geneva: CERN. URL: http://cds.cern.ch/record/2648892.